FEATURE-BASED DETECTION AND TRACKING OF INDIVIDUALS IN DENSE CROWDS

SIM CHERN-HORNG

B.Eng. (Hons) in Electrical and Computer Engineering,

National University of Singapore

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2009

Acknowledgments

My deepest thanks go to my supervisor, Assoc. Prof. Surendra Ranganath for his constant guidance, encouragement and patience throughout the years.

I would also like to thank Prof. Yedatore Venkatakrishnaiya Venkatesh, Assoc. Prof. Ong Sim Heng and Assoc. Prof. Cheong Loong Fah for their helpful comments and valuable suggestions.

I am grateful to Mr. Francis Hoon, the laboratory technologist of the Vision and Image Processing Laboratory for providing me with all the technical facilities required for these years.

I also wish to thank my colleagues and friends for always inspiring and helping me in my times of personal and financial needs.

Finally, I would like to thank my parents and my wife for their endless love and support.

SIM CHERN-HORNG

Contents

FEATURE-BASED DETECTION AND TRACKING OF INDIVIDUALS IN DENSE CROWDS

Sim Chern-Horng

Summary

Visual surveillance research is an important topic in computer vision and has received increasing attention ever since the nine-eleven attacks in 2001. Despite many existing works on detection, tracking and behavior recognition in different video surveillance environments, only a few have considered densely crowded places. This is an issue that needs to be addressed as crowded areas should be of great concern since terrorist attacks in such places can achieve maximum fatalities and provide cover for the perpetrators to escape unnoticed. This forms our motivation to detect and track people in dense crowds. Here, it is common for a person to be significantly occluded, where the visible part of a person in any camera view can be unpredictable, making it difficult to use regular windows, shapes or human models. Therefore, available methods which are human-specific model-based, region-based and contour-based are not suitable for reliably detecting and tracking individuals in this scenario.

In this thesis, we propose a feature-based approach to detect the head of an individual, which is possibly the most unoccluded part of a person in dense crowds, and track it to facilitate further processing like identification and behavior analysis. There

are no salient elements such as areas, colors and edges that can reliably represent the head of an individual in dense crowds. Therefore, we use Haar-like features to train a local head detector offline and further propose a two step post-processing procedure to improve the performance of the detector. The first step creates a color bin image from each of the initially detected windows. Every color bin image created is then classified as a correct detection or a false alarm using a trained cascade of boosted classifiers that also uses Haar-like features. The second step exploits the use of a weak perspective model for a single uncalibrated camera to further reduce the false alarm rate. This step simply relies on the 2D image size of the detected windows and their 2D locations in the image frame. However, here, we assume that the crowd is distributed over a plane and that the individuals within the crowd have the same 3D world size.

We also propose a method for tracking heads in detected windows, tailored to the scenario of dense crowds. Based on spatiotemporal measurements, our approach uses several Kanade-Lucas-Tomasi (KLT) feature points in a Bayesian framework. Here, the locations of the feature points are used to define a prior term and the motion coherency of these feature points is used to define the likelihood term. During time instants when the tracker infers a significantly occluded head, a linear approximation method is used to estimate the track. Additional characteristics of the tracker, such as robustness against scaling and rotational motions, are also proposed.

Finally, we propose a method to find the best frontal facial view of the detected and tracked person from among the multiple images in a video sequence, so as to optimize the performance of further processing, such as face recognition.

Results of the proposed head detector are presented in the form of Receiver Operating Characteristics (ROC) curves; for instance, at a 79.0% detection accuracy, the false alarm rate is 20.3%. Results of the proposed tracking system are presented qualitatively on densely crowded scenes and many other tracking scenarios, including vehicle tracking. Results of the proposed method in finding the best frontal facial view are presented with respect to person dependency, low pixel resolution, occlusion

problems and in densely crowded scenes.

List of Publications

C-H. Sim, S. Ranganath and Y.V. Venkatesh. Automatic Selection of Best Frontal Facial View from Multi-Camera Images. In *IEEE Conference on Advances in Cybernetic Systems*, September 2006. (oral presentation)

C-H. Sim and S. Ranganath. Reducing False Alarms for Detections in Crowd. In *Asian Conference on Computer Vision Workshop on Multi-dimensional and Multi-view Image Processing*, November 2007.

C-H. Sim, E. Rajmadhan and S. Ranganath. A Two-Step Approach for Detecting Individuals within Dense Crowds. In *V Conference on Articulated Motion and Deformable Objects*, July 2008. (oral presentation)

C-H. Sim, E. Rajmadhan and S. Ranganath. Using color bin images for crowd detections. In *IEEE International Conference on Image Processing* October 2008.

C-H. Sim, E. Rajmadhan and S. Ranganath. Detecting People in Dense Crowds. (in preparation)

C-H. Sim and S. Ranganath. Tracking People in Dense Crowds. (in preparation)

List of Tables

List of Figures

List of Acronyms

Center of Gravity	CG
Dynamic Time Warping	DTW
Federation International Football Association	FIFA
Field Of View	FOV
Hidden Markov Model	HMM
Histogram of Oriented Gradients	HOG
Human-Computer Interaction	HCI
Kanade-Lucas-Tomasi	KLT
Markov Chain Monte Carlo	MCMC
Multivariate Principal Component Analysis	MPCA
National University of Singapore	NUS
Pan-Tilt-Zoom	PTZ
Region Of Interest	ROI

Chapter 1

Introduction

Human-Computer Interaction (HCI), defined as the analysis of interaction between people and computers, is an interdisciplinary subject relating computer science with many other fields of study and research. For the past two decades, HCI has been an active research area in the computer vision community, encompassing smart environments, wearable computers, perceptual user interfaces and ubiquitous computing [1, 2, 3]. It represents the new generation of computing and information technology. This is because embedded devices will be everywhere, i.e. in our clothes, cars, homes, offices, roads, hospitals, or any other places one can think of.

An example of HCI is a smart environment which senses, understands and provides appropriate responses to activities happening in the environment. The research in this thesis is related to intelligent visual surveillance, a smart environment which is of topical interest in computer vision [4, 5]. Such systems monitor a scene with camera(s) and seek to understand the objects' behaviors. This is useful to replace traditional man-monitored video surveillance in places such as banks, casinos, shopping centers, streets, airports etc. because the demands greatly exceeds the capability of human operators to monitor them. Besides surveillance for security applications, it

is also applicable to patient monitoring in hospitals and care of the elderly, children and the handicapped in their homes. This is an exciting research field with excellent commercial prospects.

1.1 Visual Surveillance

In recent years, surveillance systems are being increasingly deployed in public places due to security concerns, e.g. see [6]. The ideal is to have such systems continuously operating on a "24/7" basis and provide reliable and automatic alerts to security personnel about potential threats. This section presents two real-world examples of camera-based infrastructure for surveillance, and then describes the basic processing tools that are useful in visual surveillance research.

1.1.1 Real-world Examples

Surveillance cameras are generally mounted at elevated levels in environments such as shopping malls, commercial buildings and underground subway stations. A recent article [7] estimates that there are 4.2 million surveillance cameras in Britain. However, the cameras are under utilized because of the lack of manpower to monitor and zoom into possible unusual activities happening. Thus there is an urgent need for intelligent visual surveillance systems, to increase effectiveness.

Figure 1.1 shows an area under "24/7" surveillance at the National University of Singapore (NUS), where the yellow-marked region represents the upper deck of the canteen, the red lines represent the fields of view (FOV) of each camera in its default position and the green lines represent the FOV of camera CamB in its alternative position. All cameras are Pan-Tilt-Zoom (PTZ) type, so that regions of interest can be viewed in detail by zooming in if necessary. Figure 1.2 shows typical scenes. These cameras are used for person identification, unattended object recognition or unusual

activities recognition within the monitored area. However, all these tasks are done manually.

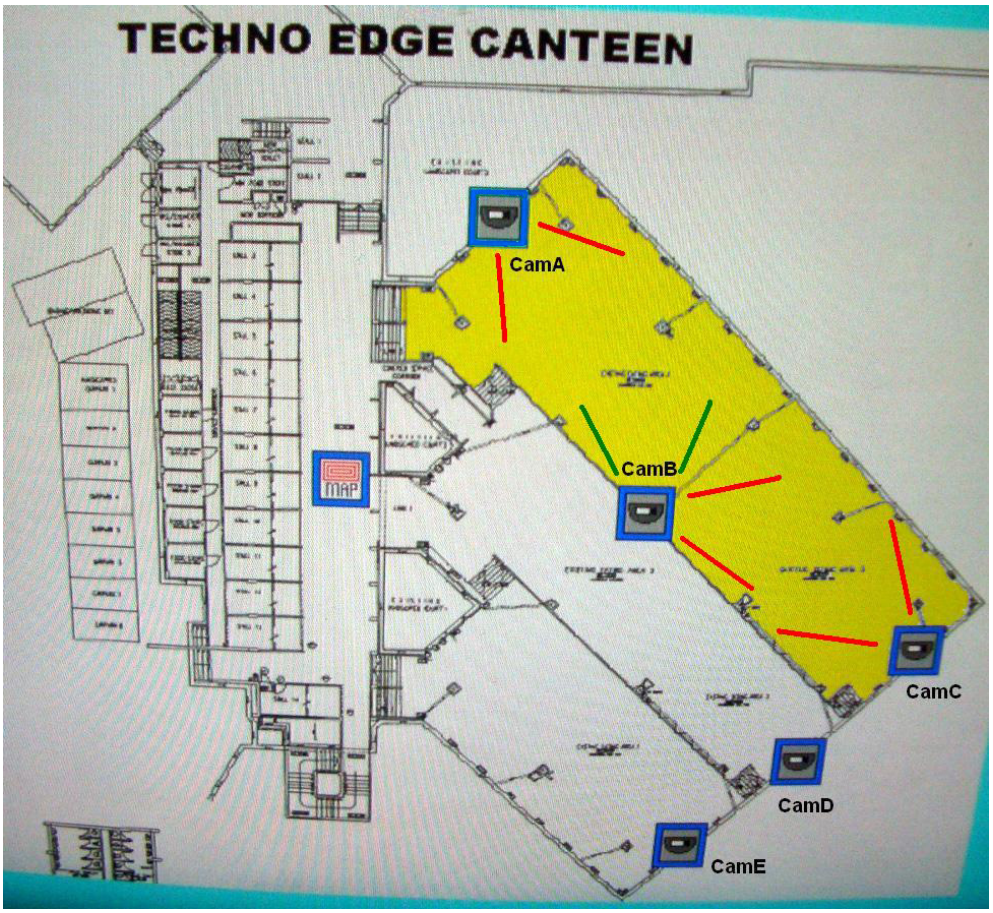
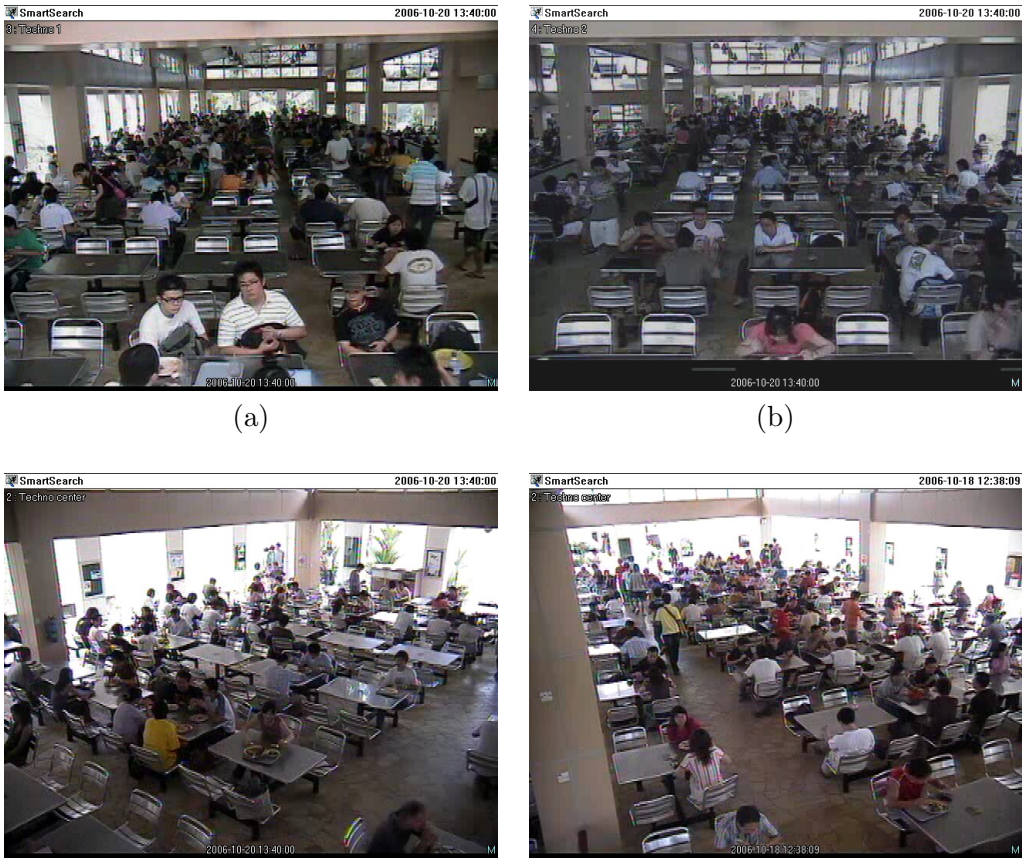


Figure 1.1: Placement of PTZ surveillance cameras in NUS Techno Edge Canteen.

1.1.2 Basic Processing Tools

As with any smart environment, visual surveillance shares the common goal of recognizing meaningful events before providing an appropriate response, e.g. reacting to a person's need in a smart environment vs sounding the alarm during an extraordinary event in a place under visual surveillance. This requires the following tasks: object



(a) (b)

(c) (d)

Figure 1.2: Surveillance at the upper deck of NUS Techno Edge Canteen: (a) Image captured by CamA at a particular time instant; (b) Image captured by CamC at the same time; (c) Image captured by CamB at the same time; (d) Image captured by CamB at another time instant in its alternative position.

detection, tracking, and behavior analysis.

Object Detection

Almost all visual surveillance systems [8, 9, 10, 11], start with object detection to obtain regions of interest (ROI) corresponding to the objects. In general, object detection methods can be spatial and/or temporal. Spatial object detection methods rely on object models using features such as points, lines or blobs; whereas temporal object detection methods use motion segmentation to obtain ROIs and then use an object classification method to identify the object.

Tracking

After object detection, surveillance systems usually track the objects, through an image sequence, before inferring object behavior. Tracking can be implemented by an object detection algorithm in every frame followed by corresponding the objects across frames as in [12], or by estimating the objects' positions in the present frame based on information from the previous and present frames as in [13]. Though the first approach (we call it a joint detection-tracking system) generally provides the objects' locations at every time instant better than the second approach (we call it an independent detection-tracking system), it is too computationally expensive for real-time video processing, for example, to process at least five frames per second.

Behavior Analysis

The next stage is to recognize and understand the activities and behaviors of the tracked objects. This can be considered as a temporal classification problem, i.e. matching a test sequence to a pre-compiled library of labeled sequences that represent prototypical actions that need to be recognized by the system. Several approaches to behavior

understanding have been used, the common ones are: dynamic time warping (DTW), a template-based dynamic programming matching technique that obtains a similarity measure between two sequences, which may vary in time or speed, [14]; hidden Markov models, a doubly stochastic model where the behavior being modeled is assumed to be a Markov process with unknown parameters, determined through training [15]; Bayesian networks, a probabilistic inference approach, using suitable factored representations of probability distributions [16]; self-organizing neural networks, which unlike the other methods use unsupervised learning and are suitable when objects' motions are unrestricted [17]. Despite all these approaches, automatic understanding of abnormal behaviors poses a big challenge due to the lack of adequate training data.

1.2 Motivation and Objective

Amongst different visual surveillance applications, surveillance of crowded areas is of great concern, since terrorist attacks in such places can achieve maximum fatality rate and provide cover for the perpetrators to escape unnoticed. However, only a few recent works [18, 19], have attempted to deal with dense crowds such as those shown in Figure 1.3. If a suspect were to mingle within such dense crowds, none of the currently available tracking methods will be reliable. This forms our motivation for intelligent visual surveillance in densely crowded places. The problem is challenging due to the dense crowd, where severe occlusions are common and limits the information that can be attained from each individual.

1.2.1 Detecting Individuals in Dense Crowds

Detection of people is the first step. Once this is achieved in dense crowds with reasonable accuracy, tracking and applications such as crowd density estimation and crowd behavior recognition (i.e. different crowd motions or unusual crowd activities) can be

(a) (b)

(c) (d)

Figure 1.3: Example images of crowds: (a) Indoor crowd with unconstrained movements; (b) Outdoor crowd without much movements; (c) Crowd exiting a common area through a doorway; (d) Crowd entering a common area through the doorway.

developed. Not many works have considered detecting individuals in dense crowds. This is much needed for visual surveillance and forms one of the two main goals of this thesis (Chapter 4).

1.2.2 Tracking Individuals in Dense Crowds

Tracking each detected person is vital to enable further processing such as identification and behavior analysis of people in dense crowds, and camera switching in a multi-camera setup. We propose to track a few individuals rather than tracking all the people in dense crowds, which is more computationally intensive. The latter is suitable for inferring crowd behavior, but our interest is from the perspective of a security official monitoring a surveillance video, where a few suspicious individuals need to be tracked, rather than all the people in the scene. No work has been done to track an individual in such dense crowds reliably, thus this forms the other main goal of this thesis (Chapter 5).

1.3 Thesis Organization

The rest of this thesis is organized as follows: Chapter 2 presents a literature review of recent works related to detection and tracking with emphasis on detection and tracking of humans in complex scenes. The chapter concludes with an overview of the proposed work, explaining the motivation for the chosen feature-based approach in this thesis. In Chapter 3, a temporal feature-based approach to detect any independent motion in an image sequence, is implemented and studied. In Chapter 4, a complete detection system is proposed for detecting individuals in dense crowds using a spatial feature-based approach. In Chapter 5, a complete tracking system based on a probabilistic approach is proposed for tracking individuals in dense crowds. The tracking system is also tested on other scenarios besides dense crowds. Chapter 6 shows an application of

the detection and tracking work in finding the best frontal facial view of a person from among the multiple images in a video sequence, for optimal further processing such as face recognition. Finally, this thesis concludes in Chapter 7 by presenting the research contributions and directions for future work.

Chapter 2

Background: Detection and Tracking

This chapter presents a literature review of both detection and tracking methods, followed by an overview of the proposed approaches to detect and track people in dense crowds.

2.1 Related Work

Here, we present several detection and tracking approaches, where the detection methods could be spatial or temporal and tracking could be a joint detection-tracking system or an independent detection-tracking system, as mentioned earlier.

2.1.1 Approaches for Spatial Detection methods

Spatial detection methods are very commonly used when information about the appearance of the object is known. These spatial methods can be categorized into four major approaches: region-based, contour-based, model-based and feature-based. Examples of

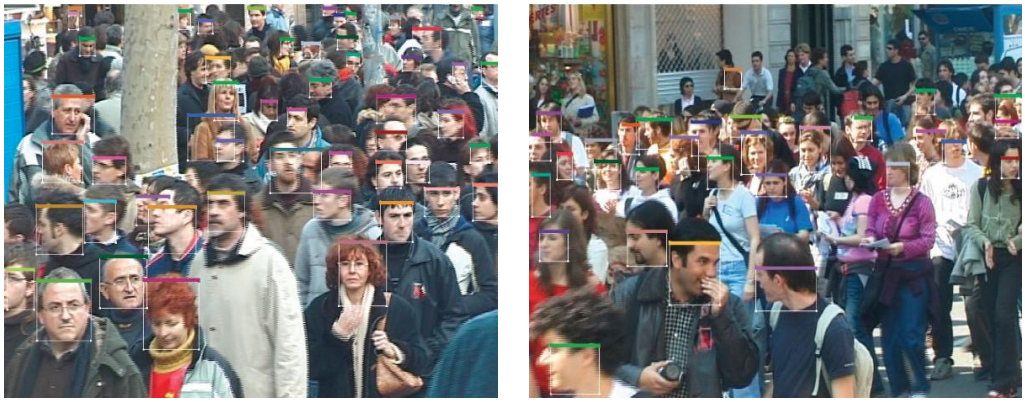
these approaches are presented in the following:

Region-based Approach

The region-based approach models the appearance of the objects and detects or tracks them according to characteristics of the image regions that correspond to the moving objects. It does not explicitly track the change in shape of the region because a rigid shape appearance model is usually assumed and used for detecting and tracking each object. It performs well in scenes containing a few objects, but it cannot reliably handle occlusion or a complex background.

Mckenna et al. [20] tackle the shortcomings of region-based approaches, as mentioned above, nicely by combining color and gradient information for segmentation and then tracking at three levels of abstraction: regions, people and groups. Each region has a bounding box that can merge and split. A human is composed of one or more regions grouped together under the geometric structure constraints of the human body, while a group consists of two or three humans grouped together. It detects and tracks region(s) while building up an individual color appearance model of each person, enabling the tracking of multiple people even during partial occlusion. In [18], Casas et al. detect individuals in crowds as in Figure 2.1 based on skin color classification and shape analysis by morphological tools. Though the method performs well under partial occlusion, it detects faces which requires people in the crowd to be facing the camera. Different from most region-based approaches which are problem-specific, Comaniciu et al. [21] apply a modified mean shift approach for region tracking. Firstly, the color, texture or edges of the selected object's region is modeled with a probability density function. Then the candidate region in the next frame is found whose density function is most similar to the object's density using the Bhattacharyya coefficient as a similarity measure. Finally, the object's density function is updated and the process continues.

Not only can it perform under partial occlusion, it can also handle instantaneous full occlusion well.



(a) (b)

Figure 2.1: Selected frames of the results from [18].

Contour-based Approach

A contour-based tracker, normally termed a "snake" or active contour, is an elastic curve that is driven to fit the detected features of an object. The fitting is done through an energy minimization procedure that draws the curve towards these features, while maintaining constraints of smoothness and continuity. One of the most commonly used features is the gradient/edge map of the image. Generally, contour-based methods detect and track objects by representing their outlines as bounding contours and updating these contours dynamically in successive frames. These algorithms aim to extract shapes of the objects directly and are able to provide more effective descriptions of the objects at a lower computational complexity, compared to region-based approaches. Moreover, they are able to track objects continuously even under noise or partial occlusion. However, the detection and tracking accuracies are limited to the 2D contour level; they are highly sensitive to the initialization, which makes it difficult to

detect the object automatically and track it.

Snakes were proposed by Kass et. al in [22], and many works have since appeared, for example, [23, 24, 25]. Paragios and Deriche [23] detect and track multiple moving objects in a sequence of images using front propagation theory and a level-set formulation. Detection and tracking boundaries are determined one after another by propagating a regular curve to each of the objects separately. A geodesic active contour objective function, defined from the detection and tracking, is then minimized using a gradient descent method and implemented for object tracking using level set theory.

Mohan et al. [24] use shapes for detection but bypass the initialization problems of using "snakes", by implementing body component detectors defined by their shapes: head, legs, left arm and right arm, and then combine them using a hierarchical classification architecture to detect a person, before tracking. In this hierarchical classifier architecture, learning of the body components using support vector machines occurs in multiple stages, with prior knowledge of the geometric structure of a human body, giving a more robust human detector than schemes which use full-body person detection methods. Peterfreund [25] proposed a new active contour model based on Kalman filter for tracking nonrigid moving objects. This approach uses gradient-based image potential along the contour as system measurements. Unlike traditional contour-based methods, this approach can track objects under instantaneous full occlusion due to the predictive nature of the Kalman filter. However, there is no work that uses contour-based approaches to detect or/and track individuals in dense crowds, due to the heavily cluttered scenes involved and thus the difficulty of initializing a "snake" on each individual.

Model-based approach

Model-based methods detect and track objects by matching projected models, which are usually constructed offline with prior knowledge of the object (such as humans or vehicles), to image data. In general, prediction strategies are used to predict the pose of the model for the next frame according to tracking history and prior knowledge of motion models and constraints. Using a similarity measure between the image data and the predicted model, search strategies are then used for finding the correct pose, which updates the predicted model, and the cycle continues. For such approaches, the more complex the human body model constructed, the more accurate the tracking result is, but the more expensive it is in computation. Thus, in comparison to other tracking techniques, these approaches are the best in dealing with occlusion problems for tracking people despite their complexity and high computational cost.

Karaulova et al. [26] use a stick figure representation to build a novel hierarchical model of human dynamics encoded using Hidden Markov models (HMMs), which is able to recover 3D skeletons in a view independent manner from monocular image sequences. $W4$ in [11] uses a combination of shape analysis and tracking to locate people and their parts (head, hands, feet, torso) and to create 2D silhouette models of people's appearance so that they can be tracked through interactions such as occlusions. 3D volumetric models such as [27] overcome the restrictions on the viewing angle of 2D models but they require more parameters leading to more expensive computation during the matching process. Lastly, to attain more accurate results, Plankers and Fua in [28] present a hierarchical human model with four levels: skeleton, ellipsoidal balls simulating tissues and fats, polygonal surface representing skin and shaded rendering.

Closer to our proposed crowded scenario, Schiele et al. [29, 30] detect individuals in crowds of at most 10 people by greedy evidence integration from different sources. Their approach combines local and global cues about a human silhouette via

a probabilistic top-down segmentation, allowing individuals to overlap or be partially occluded. Other works [31, 32, 33] by Nevatia et al., use an approach that is able to detect and track about 10 people in a video sequence with high accuracy. They use articulated ellipsoids to model human shape, color histograms to model different people's appearance and a principled Markov Chain Monte Carlo (MCMC) approach to segment individuals in a group before tracking. Figure 2.2 shows images depicting the actual crowd densities in these two works.

(a) (b)

Figure 2.2: Selected frames of the results from: (a) [29, 30]; and (b) [31, 32, 33].

Feature-based Approach

Feature-based methods detect objects by extracting salient elements, clustering them into higher level features for classification and then tracking them by matching these features between consecutive images. Traditionally, these approaches have to first initialize the features of the object when there is no occlusion. Subsequently, they are able to track the object successfully under partial occlusion or even under temporary full occlusion if velocity of the object can be distinguished and predicted accurately. Features used here may range from global features [34], such as centroids, perimeters, areas and colors, to local features [35], such as line segments, curve segments and

corner vertices. Also, features can be dependence-graph-based, and include a variety of distances and geometric relations between features, as in [36].

Jang and Choi in [37] cleverly combine the three different categories of features mentioned above. They use features like shape, texture, color, and edge to build an active template that characterizes regional and structural features of the object. Then, a feature energy function is defined and minimized during the matching process between consecutive frames, together with the use of a Kalman filter, to predict and track the object which may be nonrigid like humans. Like other feature-based approaches, this method can handle partial or temporary full occlusion if the features are initialized before any occlusion. To counter the initialization problem, Rabaud and Belongie in [19] use the Kanade-Lucas-Tomasi (KLT) feature tracker to detect and count individuals in crowds as dense as that in Figure 2.3. Their approach describes a highly parallelized version of the KLT tracker to process the video into a set of feature trajectories, which is then spatially and temporally conditioned, before being clustered with heuristic parameters to group them into individuals for counting. Results of [19] show error rates of counting individuals in dense crowds to range from 6.3% to 10.0%.

(a) (b)

Figure 2.3: Selected frames of the results from [19].

2.1.2 Approaches for Temporal Detection Methods

Compared to spatial methods, temporal detection methods are generally less effective and are normally used only if there is little or no information regarding the appearance of the object. The assumption in these methods is that regions of motion are the regions of interest. These methods are less frequently used than spatial methods, and are at times combined with spatial methods for better performance. These methods can be categorized into three different approaches: temporal differencing, optical flow and motion model.

Temporal Differencing

Temporal differencing involves subtraction of two or three consecutive frames in an image sequence followed by thresholding, to extract moving regions as object regions. However, it does a poor job of extracting all the relevant object pixels, especially if the object consists of uniform regions. This problem can be resolved if background subtraction is performed, to extract the relevant object pixels. On the other hand, the challenge of background modeling is dynamic environments which temporal differencing is very adaptive to. Hence, combining both the approaches is able to deal with both the problems as mentioned. For example, in dynamic scenes, background subtraction itself may extract irrelevant pixels foreground which temporal differencing may disregard, but it is able to assist temporal differencing in extracting the relevant object pixels. See [38].

Works that use temporal differencing are usually for detecting and tracking isolated objects and are not meant for densely crowded environments. Lipton et al. [39] use temporal differencing to detect moving objects in image sequences. The absolute difference of the current and previous frame, is thresholded and used to determine the changes. By using a connected component analysis, the changes are clustered into

motion regions. Liu et al. in [38] perform an online update to the background model and use background subtraction with frame differencing to detecting human motion.

Optical flow

Optical flow [40] is the velocity field which warps the previous frame into the current one. Motion segmentation uses characteristics of the flow vectors to detect differently moving regions in an image sequence. Optical flow approaches can be used to detect independently moving objects even in the presence of camera motion. However, they are computationally complex and very sensitive to noise, and cannot be applied to real-time video processing without specialized hardware.

Tsutsui et al. [41] present an optical flow tracking approach, computed based on the generalized gradient method, using multiple cameras in indoor environments. By exchanging information among cameras, the 3D position and velocity of the objects are estimated at all time instants. Meyer et al. in [42, 43] use optical flow to handle the initialization problem of a spatial detection method. In their proposed work, the flow field is computed to initialize a contour-based tracking algorithm, called rays, for the extraction of articulated objects. The results are then used for gait analysis. Closer to our proposed problem, mentioned works like [18, 19], also combine optical flow with other spatial approaches in detecting or counting individuals in dense crowds: [18] uses optical flow on the face candidate regions, which are extracted based on skin color classification and shape analysis, to determine faces of individuals in dense crowds; while [19] uses the flow information of the conditioned feature trajectories for clustering in order to segment individuals for counting.

Motion Model

Besides modeling a human body as mentioned earlier, motion models of human limbs and joints can also be used for detecting humans and tracking them because the movements of the limbs are strongly constrained and periodic. They are very effective as these motion models serve as prior knowledge to predict motion parameters during tracking. Motion models are usually incorporated with a model-based approach. For instance, Sidenbladh et al. [44] implement multivariate principal component analysis (MPCA) to train a walking model before using it to track a person. Similarly, Ning et al. [45] learn a motion model from acquired training data and represent it using Gaussian distributions.

2.2 Analysis for Proposed Approaches

Having reviewed the related works in detection and tracking, we see that there are several works such as, [46, 47, 29, 30] using full-body appearance models and [31, 32, 33, 48, 49] using body part detectors, that deal with sparse crowds, as in Figure 2.4. Because of possible occlusions and the cluttered environments involved, they all employ model-based approaches which are most suitable in this scenario for detection and tracking individuals. However, for the dense crowds as considered in this thesis, there are only two works [18, 19] which claim to detect, count or track each individual by using a region-based or feature-based approach, respectively with optical flow. With these works in mind, we give below our motivations to detect and track individuals in dense crowds.

Figure 2.4: Sample image frames of sparsely crowded scenes from the CAVIAR set [50].

2.2.1 Detecting and Tracking Individuals in Dense Crowds

A person in a dense crowd as in Figure 1.3 is often occluded and the visible part of the person in any camera view will be unpredictable in appearance due to occlusions and no regular windows, shapes or human models may fit even if prior knowledge of the person is available. Therefore, all human-specific model-based approaches used by [31, 32, 33, 48, 49] or [46, 47, 29, 30] are not suitable in our scenario. Also, region-based and contour-based approaches, which heavily rely on spatial information about the appearance of the object, are also not suitable for detecting and tracking individuals in dense crowds. The reason why Casas et al. [18] can detect individuals in dense crowds using a region-based approach is because they require people to be facing the camera, and hence will miss detecting individuals whose back is to the camera, such as in the scene of Figure 1.3(c). Since we impose no restriction on the orientation of people with respect to the camera, this approach is not suitable as well.

On the other hand, Rabaud and Belongie [19] use a feature-based approach where the KLT feature trajectories are used to estimate crowd density. Their approach is not quite suitable for person detection because it is unable to appropriately segment out individuals (e.g. neck, shoulder, arm, clothing etc. may be extracted even though the head of the person is visible in the image, e.g. Figure 2.5), due to inadequate recognition capability. Moreover, feature-based approaches handle occlusions and interference of unrelated structures poorly leading to reduced accuracy.

(a) (b)

Figure 2.5: Selected zoomed-in frames of the results from [19].

In conclusion, detecting and tracking people in dense crowds is a timely problem to consider in the surveillance area. Therefore, none of the currently available methods is suitable for reliably detecting and tracking individuals within dense crowds. Our approach for detection and tracking is feature-based, being the most suitable for dense crowds. Similar to the approach in [19], the detection and tracking modules are separated and independent for lower computational complexity and for real-time tracking. But unlike [19], our method aims to detect the head of an individual as it is possibly the most unoccluded part in dense crowds, and track it to facilitate further processing like identification and behavior analysis.

Chapter 3

Temporal Feature-based Approach

In Section 2.2, we concluded that our approach for detecting and tracking individuals in dense crowds, needs to be feature-based. However, as discussed, there is no salient elements such as areas, colors, edges, etc. that can reliably represent the image view of an individual in dense crowds. Nevertheless, there are useful feature points and temporal information, which were used in [19] for crowd counting. Therefore, this chapter first studies the extent to which a temporal feature-based approach can detect different moving objects in simple scenario, before discussing and finalizing our scheme for detecting and tracking individuals in dense crowds, (Chapters 4 and 5). Similar to [19], the approach here aims to use the trajectories of KLT features across frames to segment out objects, based on their independent coherent motions.

3.1 Kanade-Lucas-Tomasi Feature Tracker

The KLT tracker is chosen for its simplicity and practical effectiveness for tracking corresponding feature points between consecutive image frames. The KLT features are

small regions with more reliable features than pixel-based features. Moreover, only strong-textured features are tracked, saving on computations, as compared to pixel-based optical flow techniques.

Furthermore, in normal tracking scenarios where movements are not very quick and because we are analyzing consecutive video frames between which the lighting variation is assumed to be small, features detected in one frame are unlikely to differ in texture and position by much in the consecutive frame, unless there is a sudden occlusion. Therefore, KLT features can perform as well as any other feature descriptor available [51], under the above assumptions. Below we summarize the KLT tracking technique; see [52, 53, 54] for more details.

3.1.1 Tracking with KLT

The KLT tracker uses iterative optimization to determine motion parameters (i.e. affine, $\mathbf{E_i}$ or pure translation, $\mathbf{d_i}$) of each tracked window-based KLT feature f_i (i.e. a small rectangle of image pixels) from frame I_t to the next consecutive frame I_{t+1}, where $f_i \in \mathbf{F_t}$ and $\mathbf{F_t}$ is the space of all tracked KLT features in I_t.

Given the tracked feature f_i, the KLT tracker obtains the affine motion parameters $\mathbf{E_i}$ and $\mathbf{d_i}$ by minimizing the dissimilarity function:

$$\mathbf{e_i} = \int_{f_i} [I_{t+1}(\mathbf{E_i x_i} + \mathbf{d_i}) - I_t(\mathbf{x_i})]^2 w(\mathbf{x_i}) d\mathbf{x_i} \tag{3.1}$$

where $\mathbf{x_i} = (x_i, y_i)$, are the image coordinates of f_i and w is a weight function, usually chosen to be constant or Gaussian.

In brief, the KLT method first finds feature windows with strong textures in the first frame I_1 to compose the space $\mathbf{F_1}$. Then, KLT tracks these feature windows by finding their motion parameters and updating them in subsequent frames.

3.2 Bayesian Clustering of KLT Feature Trajectories

To use the trajectories of KLT features across frames, we first review a paper [55] which claimed to use a Bayesian clustering algorithm, an optimal technique which uses all information available, to perform detections in crowds. On closer examination, [55] seemed to be proposing a new approach to reduce the computation intensiveness in the Bayesian framework, but the details of the paper are found to be confusing and unclear.

As mentioned earlier, the aim of this chapter is to study the extent to which a temporal feature-based approach for detecting different moving objects under simple scenario setups, is successful. Therefore, similar to [55], this section proposes to use a Bayesian decision framework to group the KLT feature trajectories iteratively, in the conventional way which is computationally more expansive than the seemingly approach by [55]. The methodology of the proposed temporal feature-based approach in this section is presented in the steps below:

I) Implement the KLT tracker, as described in Section 3.1, across η consecutive image frames and initialize each moving KLT feature trajectory to be in a cluster, $c_i \in \mathbf{c}$.

II) Find the mean Euclidean distances between all possible pairs of trajectory clusters.

III) Since the clusters that are closest to each other are most likely to be from the same object, merge the two clusters which have the minimum Euclidean distance. This reduces the total number of clusters by 1. However, if the minimum Euclidean distance is more than a threshold α, do not merge them and jump to step five.

IV) Go back to step two unless the total number of clusters, $|\mathbf{c}|$, is equal to or below an experimental value β. Steps one to four are initialization steps to reduce the total number of clusters so that the computational complexities in subsequent steps are reduced. It has to be noted here that the β threshold value used should not be too small which may result in merging clusters from two different objects.

V) We first define A_{mn} as the event that clusters c_m and c_n merge and B as the measure of coherent motion of the trajectories in the two merged clusters. In this step, we aim to compute the posterior probabilities $P_{mn} = P(A_{mn}|B) = \frac{P(B|A_{mn}) \times P(A_{mn})}{P(B)}$ of all possible pairs of clusters in $\mathbf{c}$ before deciding on the merging of any clusters.

- We set $P(A_{mn}) = \frac{1}{\gamma}$ or 0, where γ is the total number of cluster pairs, whose inter-cluster distance is less than δ. This defines the prior probability for a hypothesized merging of clusters c_m and c_n, strictly based on their Euclidean distance. If they are within distance δ, $P(A_{mn}) = \frac{1}{\gamma}$, else $P(A_{mn}) = 0$. The rationale is that clusters which are far apart are unlikely to be from the same object, and hence should not be merged.

- Next, we find a measure for $P(B|A_{mn})$. First, we define Q_{uv} to be inversely proportional to the variance in Euclidean distance between two feature trajectories u and v, i.e. given two trajectories that are close, we assume that they are more likely to belong to the same object if their Q_{uv} is large. Thus $P(B|A_{mn}) \triangleq \Pi_{X_u, X_v \in (c_m \bigcup c_n)} Q_{uv} = \Pi_{X_u, X_v \in (c_m \bigcup c_n)} \frac{1}{1+Var(X_u, X_v)}$ where X_u, X_v refer to trajectories u and v. In other words, this measures the likelihood of coherent motion of trajectories in clusters c_m and c_n, given that these near clusters are merged.

- Lastly, $P(B) = \sum_{m,n}(P(B|A_{mn}) \times P(A_{mn}))$, acts as a normalizing factor to

ensure that P_{mn} is a probability that sums up to one over the whole network of possible m and n.

If $m = n$, P_{mn} is defined as the measure of intra-coherent motion of the trajectories in cluster m and the computation is simplified to $\frac{\Pi_{X_u, X_v \in c_m} \frac{1}{1+Var(X_u, X_v)}}{\sum_m (\Pi_{X_u, X_v \in c_m} \frac{1}{1+Var(X_u, X_v)})}$.

VI) Step five computes P_{mn} of all possible clusters pairs in $\mathbf{c}$. From these computed probabilities, we decide if the cluster pair, c_{m_o} and c_{n_o}, which have the highest posterior probability, $P_{m_o n_o}$, should merge. It is reasonable that if the posterior probability of merging c_{m_o} and c_{n_o} is higher than the posterior probability that they are to remain as individual clusters (i.e. $P_{m_o m_o}$ and $P_{n_o n_o}$), the merging should take place, and vice versa. The decision rule for merging is thus: if $P_{m_o n_o} \geq \lambda \times P_{m_o m_o} \times P_{n_o n_o}$ where λ is an experimental value, merge these two clusters and go back to step five and re-compute the posterior probabilities in the new $\mathbf{c}$; else repeat step six with the cluster pair that gives the next highest posterior probability. To explain this, our decision rule compares inter- and intra-coherent motions of the trajectories in the two clusters. This is independent of the value of P_{mn}, i.e. cluster pair whose P_{mn} is small might have a larger measure for inter-coherent motion as compared to their intra-coherent motions. Also, note that the value of λ should be less than 1, because the proposed definition for our posterior probability depends on the coherent motion of feature trajectories: merging of feature trajectories always results in less coherent motion, unless the features are moving rigidly which then gives the same coherent motion; in other words, $P_{m_o n_o} \leq P_{m_o m_o} \times P_{n_o n_o}$.

3.3 Experimental Results

This section studies the extent of our temporal feature-based implementation on simple scenarios, where there are at most two moving objects of interest. Tuned to suit the

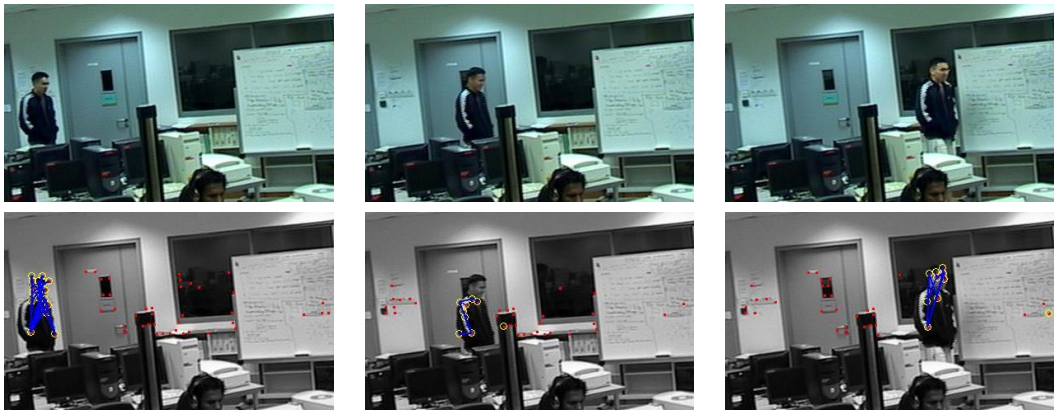
selected scenarios, the following experimental values are used based on 80 initialized KLT features: $\eta = 5$ image frames, a reasonable value to describe feature trajectories in the experimented video sequences; $\alpha = 20$ pixels, since features that are less than 20 pixels apart are unlikely to be from different objects in the experimented video sequences; $\beta = 40$, a value estimating the maximum number of objects that can occupy in each experimented scene; $\delta = 300$ pixels, since the largest object in the experimented video sequences is lesser than 300 pixels in width or height; $\lambda = 0.25$ gives the optimal performance experimentally.

Here, the KLT tracker is implemented using the public library maintained by Stan Birchfield in [56]. It took an average of 64.5 seconds to perform the motion segmentation technique on a single core 3.4GHz PC with 3GB memory. Below we present the results of object detections based on above motion segmentation for both fixed and moving camera video sequences.

Results for different fixed camera video sequences are shown in Figures 3.1 to 3.6, where the red dots refer to KLT features, encircled red dots refer to moving KLT features and the blue lines show the clustering of KLT features. Results for moving camera video sequences from [57, 58], are shown in Figures 3.7 to 3.9.

3.3.1 Results Summary

Numerical accuracy results are tabulated in Table 3.1, where detection rate refers to the ratio of the number of correct detections (i.e. clusters of KLT feature trajectories on the objects) to the number of distinct objects (i.e. people or car) in each entire video sequence, and false alarm rate refers to the ratio of the number of false detections to the sum of correct and false detections in each entire video sequence (e.g. in Figure 3.2(a), there are three clusters of KLT feature trajectories but it shows only two persons, thus resulting in one false alarm).

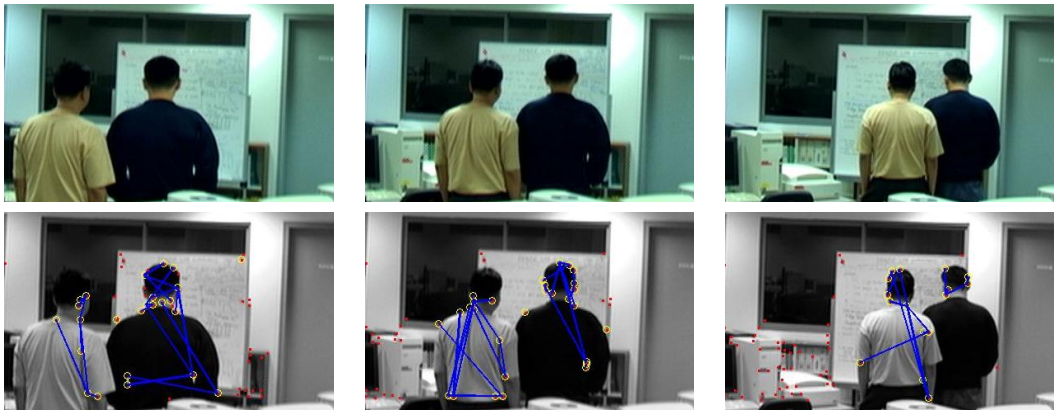


(a) (b) (c)

Figure 3.1: Person walking across the camera (upper - original image frames, lower - corresponding images showing the clustering of KLT features): (a) Interval of image frames 111 to 115; (b) Interval of image frames 186 to 190; (c) Interval of image frames 246 to 250.

(a) (b) (c)

Figure 3.2: Two persons walking towards the camera (upper - original image frames, lower - corresponding images showing the clustering of KLT features): (a) Interval of image frames 186 to 190; (b) Interval of image frames 306 to 310; (c) Interval of image frames 381 to 385.

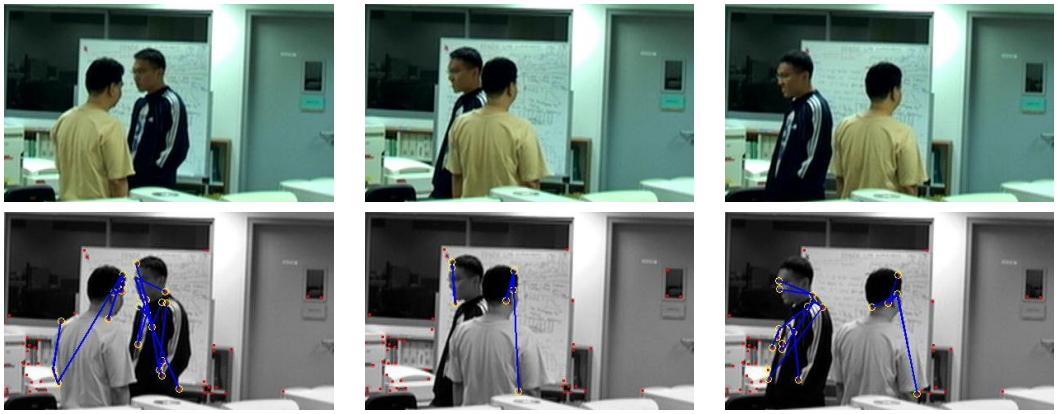


(a) (b) (c)

Figure 3.3: Two persons walking away from the camera (upper - original image frames, lower - corresponding images showing the clustering of KLT features): (a) Interval of image frames 126 to 130; (b) Interval of image frames 201 to 205; (c) Interval of image frames 306 to 310.

(a) (b) (c)

Figure 3.4: Two persons meet and leave together (upper - original image frames, lower - corresponding images showing the clustering of KLT features): (a) Interval of image frames 101 to 106; (b) Interval of image frames 138 to 142; (c) Interval of image frames 390 to 394.



(a) (b) (c)

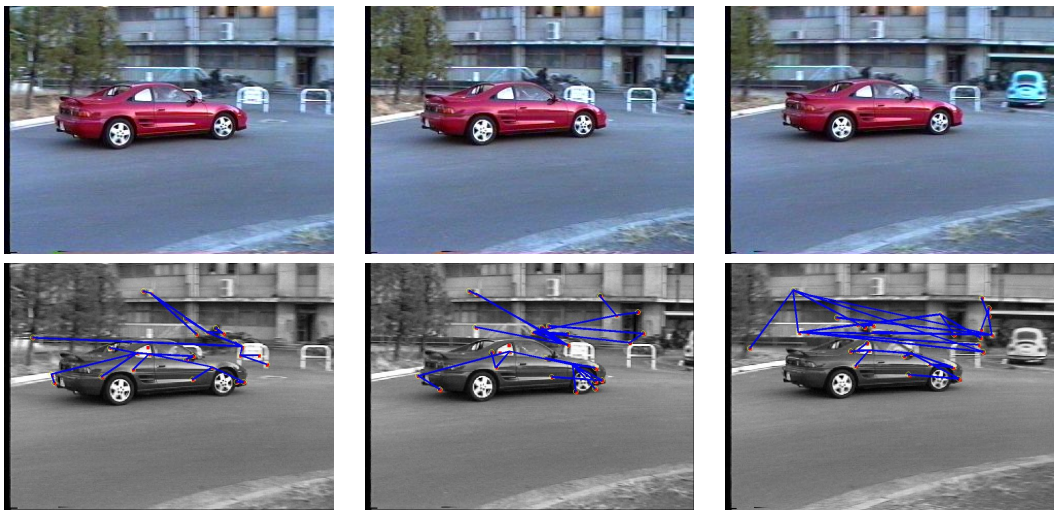
Figure 3.5: Two persons walking in opposing directions (upper - original image frames, lower - corresponding images showing the clustering of KLT features): (a) Interval of image frames 150 to 154; (b) Interval of image frames 168 to 172; (c) Interval of image frames 180 to 184.

(a) (b) (c)

Figure 3.6: Two persons walking across the camera (upper - original image frames, lower - corresponding images showing the clustering of KLT features): (a) Interval of image frames 114 to 118; (b) Interval of image frames 138 to 142; (c) Interval of image frames 156 to 160.

(a) (b) (c)

Figure 3.7: Car moving off from a car park (upper - original image frames, lower - corresponding images showing the clustering of KLT features): (a) Interval of image frames 101 to 105; (b) Interval of image frames 107 to 111; (c) Interval of image frames 125 to 129.



(a) (b) (c)

Figure 3.8: Car turning a corner (upper - original image frames, lower - corresponding images showing the clustering of KLT features): (a) Interval of image frames 98 to 102; (b) Interval of image frames 103 to 107; (c) Interval of image frames 108 to 112.

Video Sequence	*No. of Correct Detections*	*No. of Missed Detections*	*No. of False Detections*	*Detection Rate(%)*	*False Alarm Rate(%)*
Person walking across the camera as in Fig. 3.1 (85 video frames)	15	2	1	88.24	6.25
Two persons walking towards the camera as in Fig. 3.2 (95 video frames)	38	0	6	100.00	13.64
Two persons walking away from the camera as in Fig. 3.3 (85 video frames)	31	3	4	91.18	11.43
Two persons meet and leave together as in Fig. 3.4 (100 video frames)	88	12	8	88.00	8.33
Two persons walking in opposing directions as in Fig. 3.5 (90 video frames)	35	1	1	97.22	2.78
Two persons walking across the camera as in Fig. 3.6 (65 video frames)	22	4	0	84.62	0.00
Car moving off from a car park as in Fig. 3.7 (25 video frames)	10	0	0	100.00	0.00
Car turning a corner as in Fig. 3.8 (20 video frames)	8	0	0	100.00	0.00
Person rotating his head while camera moves as in Fig. 3.9 (65 video frames)	23	3	1	88.46	4.17
OVERALL	270	25	21	**91.53**	**7.22**

Table 3.1: Summary results of the motion segmentation algorithm.

(a) (b) (c)

Figure 3.9: Person rotating his head while camera moves (upper - original image frames, lower - corresponding images showing the clustering of KLT features): (a) Interval of image frames 33 to 37; (b) Interval of image frames 63 to 67; (c) Interval of image frames 75 to 79.

For the detection rates of the fixed camera video sequences, it is seen that only two of them, i.e. *two persons walking towards the camera* and *two persons walking in opposing directions* in Figures 3.2 and 3.5 respectively, achieved accuracy above 97.0%. This is because the objects in these two video sequences are higher in resolution in comparison to the other video sequences, see Figure 3.10, and they are frontal to the camera, hence providing the KLT tracker with stronger-textured feature points to track, than in the other video sequences. Unlike these two videos, the objects in *person walking across the camera*, *two persons walking away from the camera* and *two persons meet and leave together* in Figures 3.1, 3.3 and 3.4 respectively, are lower in resolution, see Figure 3.10; furthermore, Figures 3.3 and 3.4 show that the objects are mostly non-frontal to the camera, and thus possibly achieve below 92.0% accuracy. Finally, the video sequence *two persons walking across the camera* in Figure 3.6 has the worst performance of 84.62% because one of the two objects is significantly occluded for quite

some time which is much longer than the occlusion time period in *two persons walking in opposing directions* of Figure 3.5. If the missed detections due to occlusion are not taken into account for the video sequence in Figure 3.6, the result is as high as 95.0%. This is because the objects of this video sequence are also relatively higher in resolution as in Figures 3.2 and 3.5.

The false alarm rates of the fixed camera video sequences appear to be quite high even for these simple scenarios. The reason is because this temporal detection method relies on the motion coherence of KLT feature trajectories which has no capability of recognizing the object spatially, and hence articulations of the human body such as the movements of the limbs are also taken as new detections, resulting in high false alarm rates. Specifically, the false alarm rates of video sequences *two persons walking towards the camera*, *two persons walking away from the camera* and *two persons meet and leave together* in Figures 3.2, 3.3 and 3.4 respectively, attain higher false alarm rates of more than 8.0% because the objects undergo significant scaling motions, but the KLT feature trajectories clustering approach here only considers translational motions.

On the other hand, it can be seen from Table 3.1 that for rigid objects like car, the method works perfectly despite the moving camera. And for the moving camera video sequence *person rotating his head while camera moves* in Figure 3.9, the explanation of its result is that the head motion consists of significant rotational motions but the clustering algorithm here only considers translational motions, as mentioned earlier.

In summary, this non-object specific motion-based detector yields results with a high detection rate of 91.53%, with 7.22% false alarm rate. It can therefore be concluded that this detector is able to perform well in the above scenarios or other similar scenarios, whether the camera is stationary or not, even with no prior knowledge about the object(s) of interest under cluttered background. However, unlike other

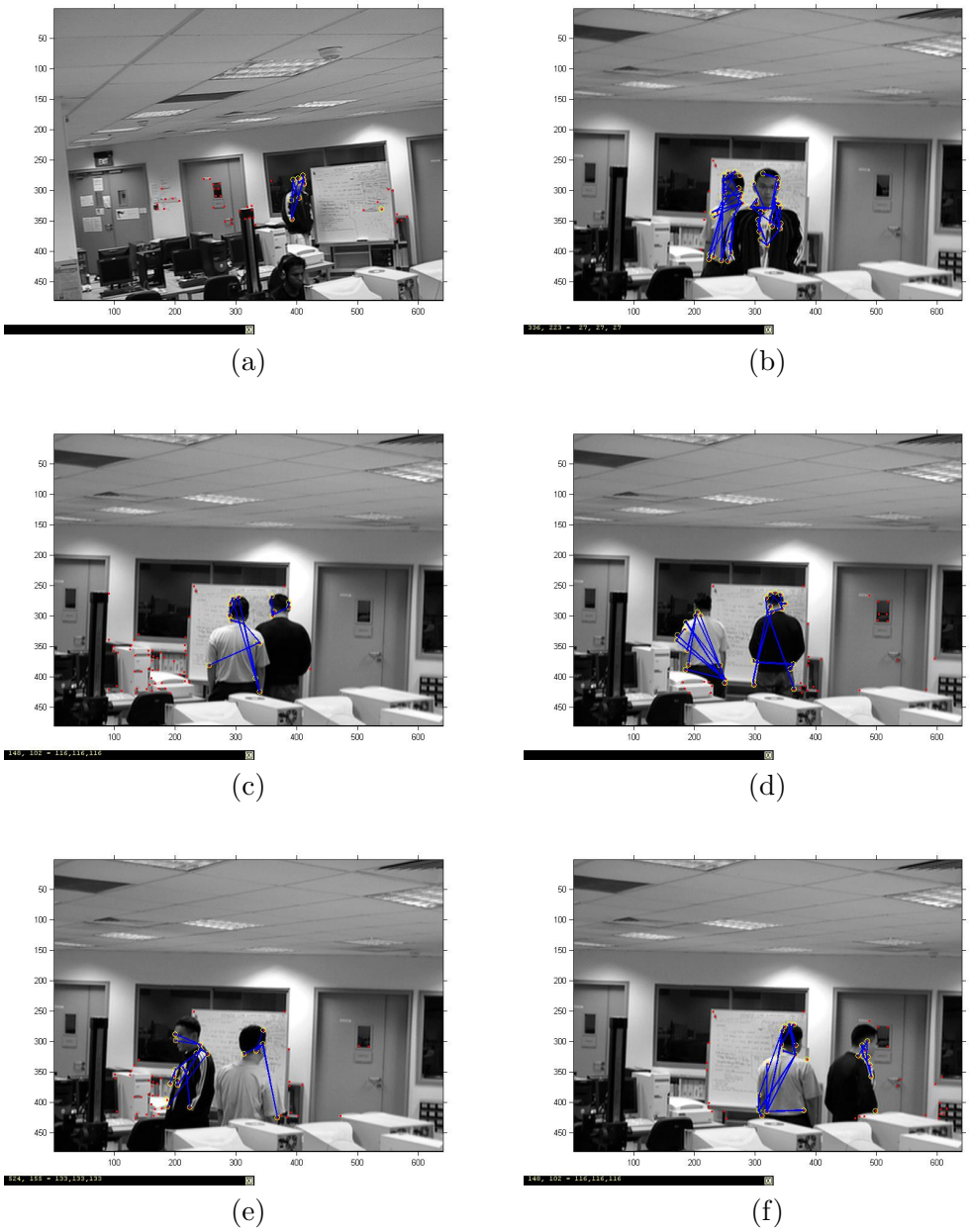


Figure 3.10: Original uncropped images of: (a) Figure 3.1(c); (b) Figure 3.2(c); (c) Figure 3.3(c); (d) Figure 3.4(c); (e) Figure 3.5(c); (f) Figure 3.6(c).

detection approaches as described in Section 2.1, the performance of this detector depends much on the rigidity of the object to be detected and it assumes translational object motions.

3.4 Concluding Discussion

Prior knowledge such as colors, shapes, sizes and models of the objects of interest is not often available, especially for surveillance systems. Also, objects of interest are in general not stationary. This motivated us to experiment with a motion-based object detection algorithm which does not require any prior information, but assumes that different objects move at different instantaneous velocities rigidly.

In the experiments presented here, most video sequences are simulations of simple scenarios (e.g. Figures 3.1 to 3.6 show people moving with no natural articulations of the heads or limbs; Figure 3.9 shows a face that is expressionless throughout the whole video sequence). However, nonrigid objects rarely move rigidly and nonrigid motions can be the main cause for poor performance in such an approach. Moreover, just like any other feature-based approach or specifically [19], some of the detected objects also exclude important parts of the object (e.g. the head is not part of the detected object in Figure 3.1(b); the front portion of the car in Figure 3.7(b) is not detected). The reason is because these important portions of the objects are not recognized by the KLT tracker as they do not consist of any strong-textured features to be tracked.

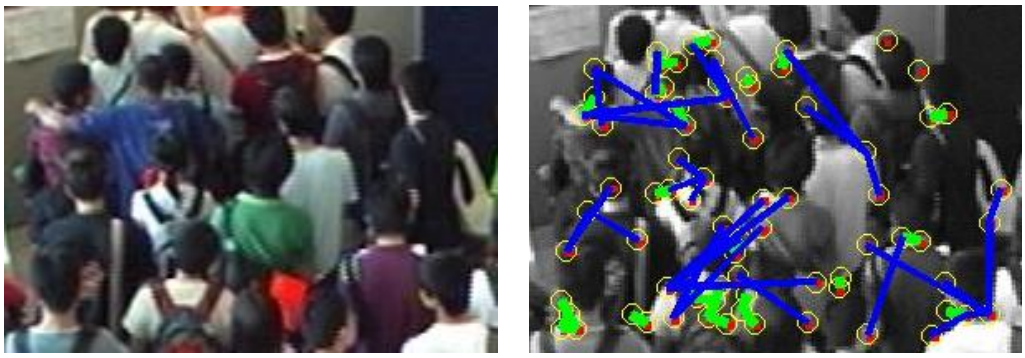
Extending the problem to surveillance scenes of crowds, the assumption that each person in the crowd moves rigidly and independently is necessary. Figure 3.11 presents the result on a sparsely crowded scene, while Figure 3.12 shows the result on a similar scene but with a denser crowd. These results are from the same approach that attained reasonably good performance on our previous video sequences, where the following experimental values were tuned to: $\eta = 9$ image frames, $\alpha = 20$ pixels,

$\beta = 20$, $\delta = 100$ pixels, $\lambda = 0.25$ and the number of KLT features initialized is 40, to best suit both scenes. Though the sparsely crowded scene in Figure 3.11 gives a perfect detection result with no false alarm, a more crowded scene as in Figure 3.12 yielded disappointing results where only 4 out of 18 persons were detected and the other detections were either false alarms or merging errors (i.e. clustering the KLT features on two or more persons). This is because the assumption that each person in the dense crowd moves rigidly and independently cannot be made. Hence it can be said that motion-based approach is unsuitable for detection of people in dense crowds.

(a) (b)

Figure 3.11: Showing motion-based detections on a sparsely crowded scene: (a) Original image; (b) Corresponding image showing clustering of KLT features.

Concluding this discussion, temporal feature-based detection approach has its strengths in non-object specificness, being able to handle cluttered or complex backgrounds and moving camera, if the objects of interest are in rigid motion. Despite in dense crowds, it is still too stringent to assume human motions to be fully rigid because of the combined visible head and shoulder articulations. This implies that the temporal feature-based detection approach is not suitable for detecting or segmenting people in dense crowds. Therefore we look into the possibility of using a spatial feature-based detection approach in the next chapter, so that important portions of the people in



(a) (b)

Figure 3.12: Showing motion-based detections on a densely crowded scene: (a) Original image; (b) Corresponding image showing clustering of KLT features.

dense crowds, which can aid further processing like face recognition or behavior analysis, are not neglected in the detections. Nevertheless, the temporal feature-based approach might still be a suitable approach for tracking individuals in dense crowds where no regular windows, shapes or human models may describe. However, this can be made possible only when good initializations are available to the tracker (i.e. just like a non-object specific mean shift tracker which uses the current image window as a reference or template to estimate the next position). In other words, such a temporal feature-based tracker (or even the mean shift tracker) will not be able to track the object if initialization of the tracker is not on the object or consists of too much noise (i.e. pixels of background or occluding objects). We will investigate this more in Chapter 5.

Chapter 4

Detecting Individuals in Dense Crowds

In this chapter, we consider a spatial feature-based detection approach to detect the head region of people in dense crowds, and thereby facilitate further downstream processing like face recognition or behavior analysis. Here, we first train a local head detector and propose a method to increase the detection rate of this detector while maintaining its false alarm rate, or alternatively to reduce the false alarm rate while maintaining the detection rate. The method follows a two step procedure. The first step relies only on color information within the initially detected windows. In every detection window, the normalized rg histogram is formed and mapped to a representative image (called color bin image). Each such color bin image is then classified as a correct detection or as a false alarm using a trained cascade of boosted classifiers using Haar-like features. The second step exploits the use of a weak perspective model for a single uncalibrated camera to further improve the performance of the system. In contrast to [59], which requires the camera height and horizon line of the 3D world, we simply rely on the 2D image size of the detected windows and their 2D locations in the

image frame. However, we assume that the crowd is distributed over a plane and that individuals within the dense crowd have the same 3D world size [60].

4.1 Haar Cascade Head Detector

We train a Viola-Jones object detector [61], which is a state-of-the-art object detector, to initially detect heads in crowds. This detector has been used in many works [46, 62, 63] to detect humans, though not within crowds, and it has also been found to be competitive with other pedestrian classifiers in a recent study [64].

We choose to build a head detector, because the head is the most prominent part of an individual and least likely to be occluded in a dense crowd; limbs and torso are usually occluded, as seen in Figure 1.3. Though head detectors in the literature such as [65, 66, 67, 68, 69] are robust with respect to head pose, they usually focus on less cluttered scenes as in Figure 4.1, where the elliptical contour of a head can be easily extracted by motion segmentation or background subtraction. On the other hand, we aim to detect heads at resolutions as small as 20 x 20 pixels in single images from much more cluttered scenes, as shown in Figure 1.3. Here the gradient/edge map can be complex, making it very challenging to automatically extract elliptical contours of heads. These considerations motivated us to use the approach of [61] to detect heads in such scenes. We also experimented with another local head detector using the approach of [70], based on the Histogram of Oriented Gradients (HOG), and compared results with the Viola-Jones head detector in Section 4.3.

Training the Detector

The object detector developed by Viola and Jones [61], and later extended by Lienhart and Maydt [72], uses a rich dictionary of simple appearance filters, similar to Haar wavelets (Figure 4.2) and a cascade of boosted tree classifiers. At each stage of the

(a) (b)

Figure 4.1: Sample images of head detection scenarios: (a) A simple scene of two people walking at different speeds; (b) Another scene where heads to be detected are higher in resolution [71].

cascade, a weighted combination of simple classifiers is iteratively constructed, until a certain user-defined performance target is met, and then the training process continues with the next stage of the cascade. Examples of positive and negative training samples for head detection are shown in Figure 4.3.

Initial Results

Figure 4.4 shows example results from this initial detector on two test images where each circle represents a detection. The test images used were different from the images used for training. In object detection problems, requiring high detection rates also leads to an increase in false alarm rates. For example in Figure 4.4, it can be seen that almost every individual in the dense crowd has been detected, but the number of false alarms is also significantly high (false alarm rate of 35.7% at a detection accuracy of 87.8%).

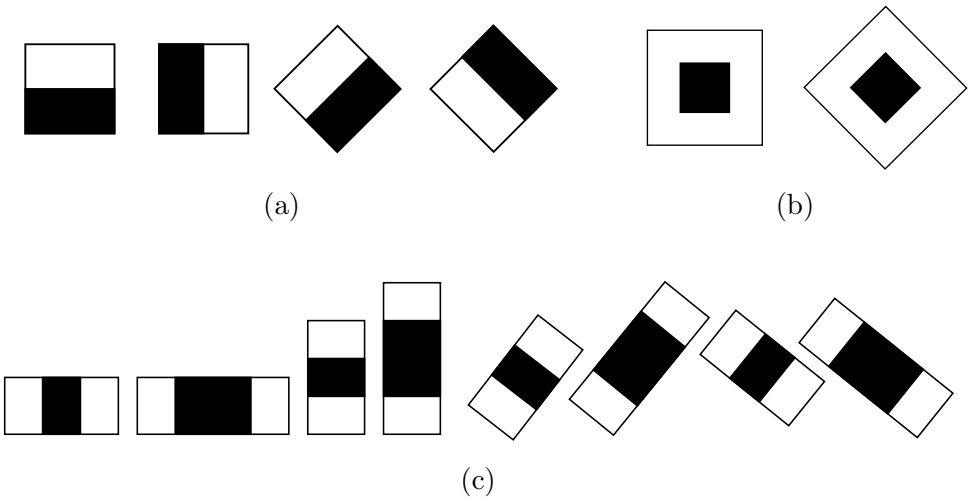


Figure 4.2: Examples of Haar-like features [72]: (a) Edge features; (b) Center-surround features; (c) Line features.

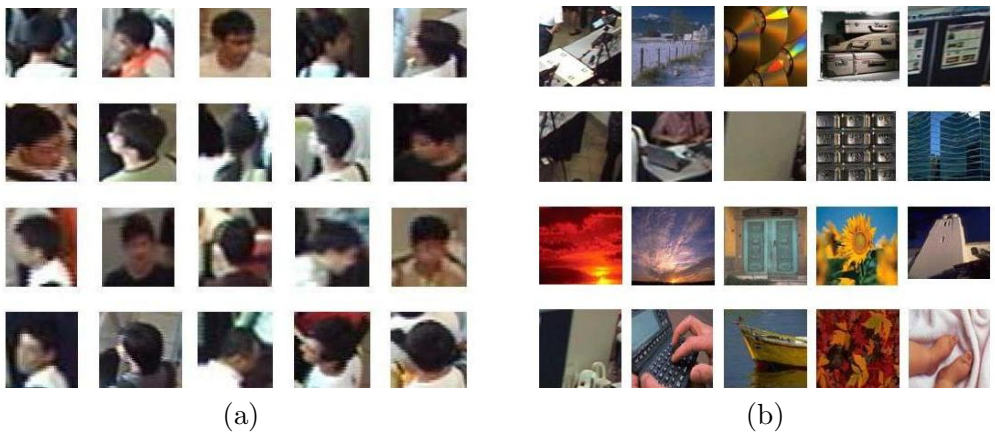


Figure 4.3: Training image examples for the local detector: (a) Positive (head) samples; (b) Negative (head) samples.

(a) (b)

Figure 4.4: Example images of dense crowds showing detections with the Viola-Jones head detector: (a) Detections on Figure 1.3(c); (b) Detections on Figure 1.3(d).

4.2 Reducing False Alarms

In this section, we take advantage of the high detection rates of the trained Viola-Jones head detector, but seek to reduce the false alarms as much as possible. We propose two independent methods for this purpose. In Section 4.2.1 we propose an alternative representation for the color information in the initially detected windows and use it to classify the windows as true detections or false alarms. In another method described in Section 4.2.2, we use properties of the scene geometry to further reduce false alarms.

4.2.1 The Color Bin Image Approach

Here, we propose a representation called the color bin image which is extracted from the initially detected windows and use it for training a classifier to improve the performance of the initial detector.

Color Analysis of Detected Windows

Every correctly detected window from the initial detector will contain a human head, i.e. will consist mainly of skin and/or hair color. Figures 4.5(a) and 4.5(b) show the normalized average RGB histograms of all the positive and negative training samples, respectively, obtained from the original training samples as in Figure 4.3. Figures 4.5(c) and 4.5(d) show corresponding RGB histograms when the pixels are linearly weighted by a factor p_{ij} defined as

$$p_{ij} = (\lfloor w/2 \rfloor + 1 - |i - \lfloor w/2 \rfloor|) \times (\lfloor h/2 \rfloor + 1 - |j - \lfloor h/2 \rfloor|) \tag{4.1}$$

where i and j are the column and row indices of the window, respectively, and w and h are the width and height of the window. Here, pixels towards the center of the window receive higher weights p_{ij}, than those towards the border.

For the negative training samples, Figures 4.5(b) and 4.5(d) appear very similar, despite the pixel weighting used for Figure 4.5(d). However, in the case of positive training samples, Figure 4.5(c) shows that the RGB intensity values in the 75-200 range have been suppressed by pixel weighting, compared to the unweighted pixel histograms in Figure 4.5(a). These suppressed intensity values are likely to be from the periphery of the positive training windows (see Figure 4.3(a)), where skin and/or hair color may not usually be found. Hence, the pixel weighting can be viewed as a means to improve the classifiability of the positive training samples.

Note that even though the weighted histograms in Figure 4.5(c) have suppressed tails, conventional thresholding or maximum likelihood classification will not be able to effectively separate the positive and negative samples, as there is considerable overlap in the histograms of Figures 4.5(c) and 4.5(d). Hence, we propose an alternative representation called the color bin image, to improve the classification.

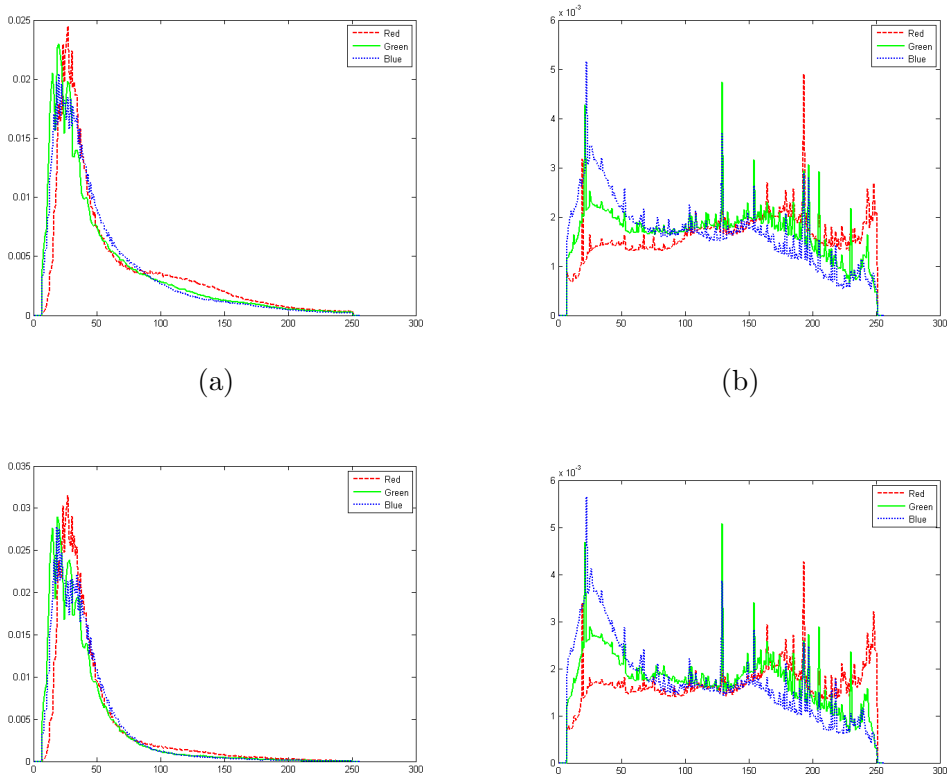


Figure 4.5: Normalized average *RGB* histograms from the original training samples: (a) Positive samples with uniform pixel weighting; (b) Negative samples with uniform pixel weighting; (c) Positive samples with pixel weighting p_{ij}; (d) Negative samples with pixel weighting p_{ij}.

Color Bin Images and Classification

Even though the histograms of the two classes overlap, their shapes are different, and we capitalize on this observation to derive a more discriminatory representation. For example, if we consider the 1D Haar wavelet transform basis functions shown in Figure 4.6, we notice that $\int f_2(u)g_{rgb}(u)du$, where $f_2(u)$ denotes Function 2 in Figure 4.6, and $g_{rgb}(u)$ denotes any of the RGB histograms in Figure 4.5(c), is much larger than zero, but is close to zero if $g_{rgb}(u)$ is any of the RGB histograms in Figure 4.5(d). This suggests that this feature can be used as a weak classifier for the two classes. Likewise, we may consider each of the Haar wavelet basis functions as in Figure 4.6 to implement a weak classifier, and use these to build a boosted classifier.

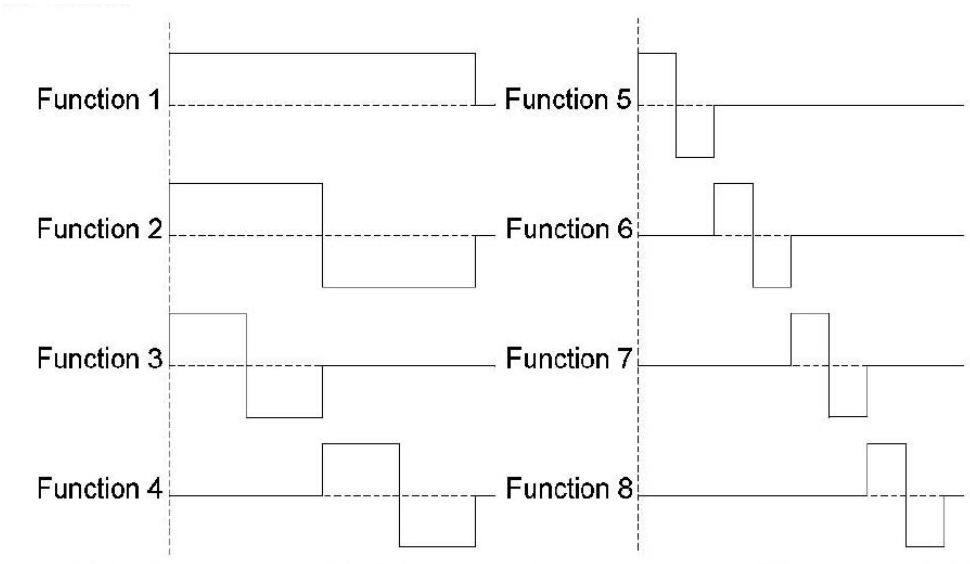
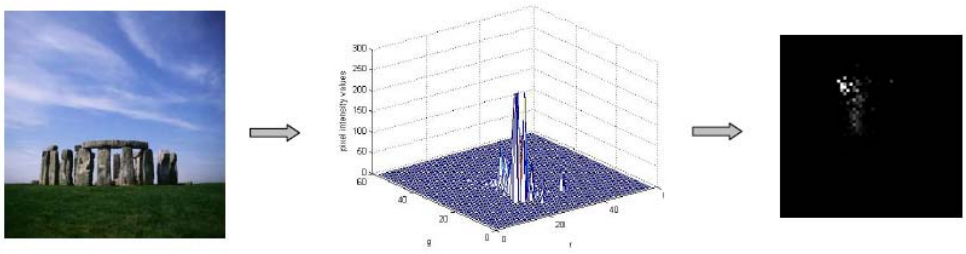


Figure 4.6: Basis functions of the Haar wavelet transform.

Rather than using the 3D RGB histograms, we use the normalized 2D rg histograms to save computations and reduce sensitivity to illumination changes. To construct a color bin image, we first obtain the normalized rg representation of the pixels in the detected window and weight the pixels with p_{ij} as in Equation (4.1), where

$r = R/(R+G+B)$, $g = G/(R+G+B)$ (and $b = 1-r-g$). Then, the rg histogram of the pixels is estimated. This 2D array is converted to a 20 x 20 color bin image where the peak value of the histogram is mapped to the highest pixel intensity in the image and the r and g axes represent the height and width of the image, respectively. An example is shown in Figure 4.7.



(a) (b) (c)

Figure 4.7: Creating a color bin image: (a) Original image; (b) 2D rg histogram of original image; (c) Corresponding 20 x 20 color bin image.

We extract Haar-like features from the 2D color bin images using the templates shown in Figure 4.2. The color bin images were obtained from the original training samples such as those in Figure 4.3. These features are used to train a boosted Viola-Jones type classifier as in [61].

Proposed Algorithm using Color Bin Images

The detected windows from the first pass using the Viola-Jones type head detector are cropped out and each of them is transformed to the color bin image representation. These color bin images are classified in a second pass as true detections or false alarms with the trained color bin image classifier. This process is depicted in Figure 4.8, and Figure 4.9 shows example results from the detection system at this stage where each circle represents a detection.

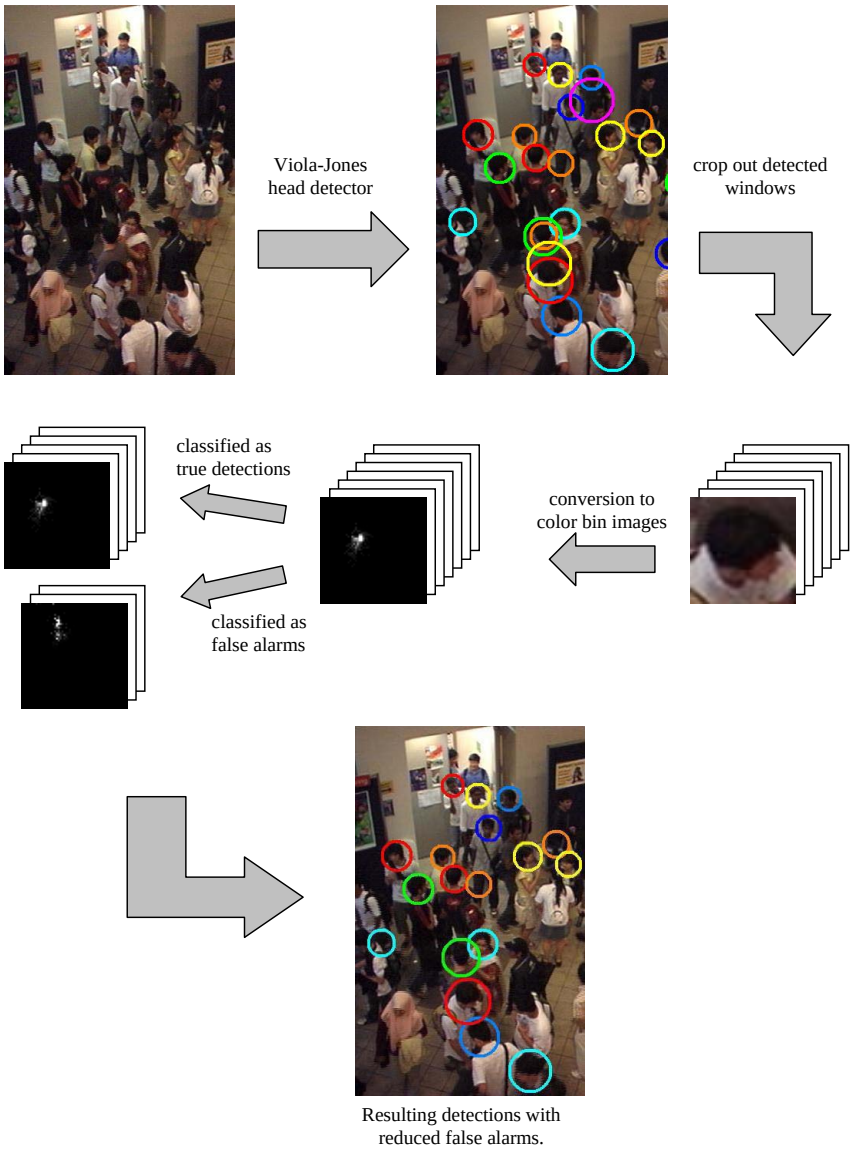


Figure 4.8: Head detection process with an initial pass of the Viola-Jones head detector followed by the classification using color bin images.

(a) (b)

Figure 4.9: Example images of dense crowds showing resultant detections using the color bin image approach (refer to Figure 4.8): (a) Resultant image where Figure 1.3(c) is the input image; (b) Resultant image where Figure 1.3(d) is the input image.

4.2.2 Regression Line Approach

Examination of Figure 4.9 shows that even though a large number of false alarms have been removed in the second pass, many are still present. Here, we propose another method to further reduce the false alarm rate while maintaining the detection rate from the second pass.

Analysis Of Detected Window Sizes

From Figure 4.9, it can be seen that the largest and the smallest circles are likely to be false alarms. Also, we notice that the detection circles become smaller towards the top of the image and vice versa (the origin is at the upper left corner of the image and the y or the vertical axis points down). Based on this observation, we seek to find a relationship between the detection circle size and its vertical location in the image.

Figure 4.10 shows the geometry of scene projection to infer image size of the person, and his location on the vertical axis of the image. Under a weak perspective camera model, we can write:

$$\frac{S_{image}}{S_{world}} \approx \frac{f}{z} \tag{4.2}$$

and

$$\frac{D_{image}}{D_{world}} \approx \frac{f}{z} \tag{4.3}$$

where:

f	camera focal length
z	depth
S_{image}	image size
S_{world}	world size
D_{image}	image distance
D_{world}	world distance

From Equations (4.2) and (4.3), we have

$$D_{image} \approx S_{image} \times (D_{world}/S_{world}) \tag{4.4}$$

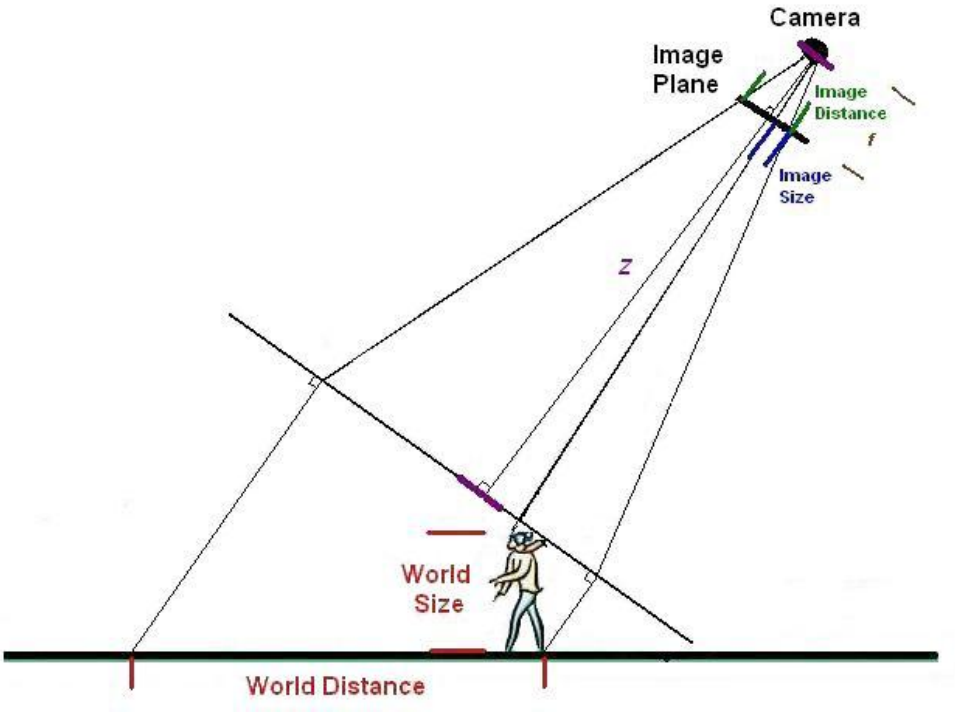


Figure 4.10: Projection of a person and world distance.

If we assume that the individuals in the crowd have the same 3D world size

and that the crowd is distributed on a plane, D_{image} and S_{image} are approximately linearly related. This implies that the sizes of the detection circles and their vertical image locations should be approximately linearly related (see also [73] for an alternative way to represent this linear relationship). Hence, in Figure 4.9, we may expect this approximate linear relationship to hold, and use it as the basis to remove false alarms as outliers. The assumptions we made lead to this novel result which does not require any 3D information as in [59], where camera height and horizon line of the 3D world are needed.

Outlier Removal using Least Squares Line Fitting

Usually, outliers are detected after fitting the best model/line to the data with methods as in [74, 75]. In some works, outliers are removed using the forward search approaches [74], where a small subset of the data is initially selected and observations are sequentially added. These methods perform effectively in removing outliers, even when the number of outliers is comparable to the number of true data points, as long as the actual data distributes linearly. In our case, the linear relationship between vertical image coordinates and size of detected windows is only approximate due to the assumptions made using a weak perspective camera model. Hence, using these methods, which assume a strictly linear relationship, could remove part of the actual data in our case. Our algorithm, presented below, is therefore a backward search approach where the entire set of observations is initially considered and the outliers are sequentially removed. It is based on the correlation coefficient of least squares line fitting, and is used for its simplicity and practical effectiveness.

The detection circles obtained as in Figure 4.9 are represented by their vertical image coordinate, y, and diameter, d. An example plot of y vs d is shown in Figure 4.11. We propose an iterative method to remove outliers based on the correlation coefficient

r of least squares line fitting given by

$$r = \frac{n(\sum dy) - (\sum d)(\sum y)}{(\sqrt{n(\sum d^2) - (\sum d)^2})(\sqrt{n(\sum y^2) - (\sum y)^2})} \tag{4.5}$$

where n is the number of detections.

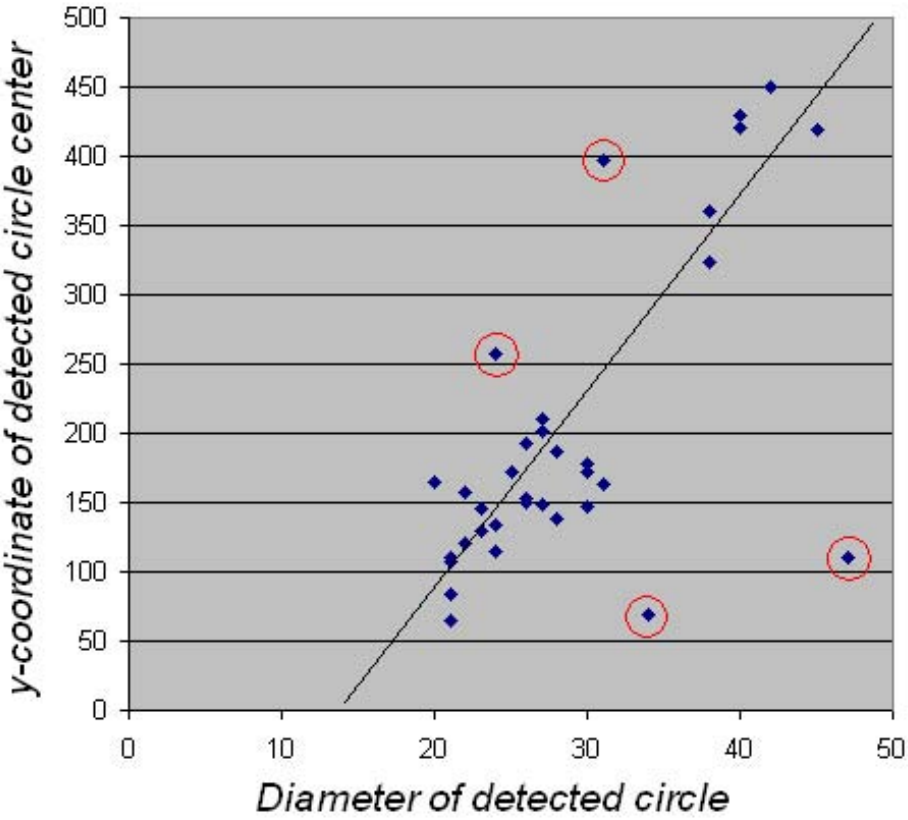


Figure 4.11: Example scatter plot of vertical image coordinate of detection circle centers, y, vs. corresponding circle diameters, d, in an image of dense crowds (the circled plots are the detected outliers which are true false alarms).

This measures the quality of line fit to the data, and we use it for removing outliers or our presumed false alarms (the circled plots in Figure 4.11).

Our algorithm detects and removes the most likely outlier in an iterative process,

using the detections in a leave-one-out scheme, and calculating the resulting correlation coefficients (for n detections, n correlation coefficients are calculated). Since a higher value of correlation coefficient implies a better line fit to the data, the left-out detection which yields the maximum correlation coefficient is the most likely outlier in each iteration and is removed from the detected set. Since the magnitude of a correlation coefficient ranges from 0 to 1, these iterations loop until the minimum magnitude of all correlation coefficients is at least 0.5. Algorithm 1 gives the pseudocode for the procedure.

Algorithm 1 Removing false alarms with regression line approach.

for all $i \in \{1, 2, ..., n\}$ **do** ▷ r_i initializations.
$m \leftarrow n - 1$
$r_i \leftarrow \frac{m(\sum dy) - (\sum d)(\sum y)}{(\sqrt{m(\sum d^2) - (\sum d)^2})(\sqrt{m(\sum y^2) - (\sum y)^2})}$
▷ where all summations exclude the i^{th} detection.
end for

while $min\{|r_i|\} < 0.5$ and $n > 1$ **do**

remove k^{th} *detection where* $|r_k| = max\{|r_i|\}$
$n \leftarrow n - 1$
for all $i \in \{1, 2, ..., n\}$ **do**
$m \leftarrow n - 1$
$r_i \leftarrow \frac{m(\sum dy) - (\sum d)(\sum y)}{(\sqrt{m(\sum d^2) - (\sum d)^2})(\sqrt{m(\sum y^2) - (\sum y)^2})}$
▷ where all summations exclude the i^{th} detection.
end for

end while

The algorithm first computes n correlation coefficients, r_i, each obtained by leaving out the i^{th} detection. If all the n correlation coefficients so computed satisfy $|r_i| \geq 0.5$, then none of the detections is removed as a possible outlier (false alarm). If however, one or more of the $|r_i|$ is smaller than 0.5, we take it as an indication that

outliers are still present, and we remove the k^{th} detection corresponding to $\max |r_k|$. The process is repeated with $n-1$ samples, and continues iteratively until for all remaining detections, $|r_i| \geq 0.5$.

4.3 Experimental Results

4.3.1 Experimental Setup

For detecting individuals in dense crowds, where our focus is more on increasing the detection rate or reducing the false alarm rate, we first consider an indoor environment with a camera mounted at an elevation and an Asian crowd with the crowd density as in Figure 1.3. The following describes the training of the local head detectors with the Viola-Jones approach, and the Dalal and Triggs approach [70] (which we use for comparison), where both the training and testing images are obtained from the same mounted camera but at different time instants.

To train the Viola-Jones head detector presented in Section 4.1 and the classifier in Section 4.2.1, we used the Open Source Computer Vision Library [76] implementation, and set the target performance for each stage to a minimum detection rate of 0.999 at a maximum false alarm rate of 0.5. To train the detector, we manually cropped a total of 4016 images of human heads captured at arbitrary visual angles against complex backgrounds, and 6816 images which do not contain any human heads, as positive and negative training samples, respectively. Though this is a reasonable number of positive and negative training samples, the positive training samples were further expanded to include the lateral inversions of all the original 4016 positive training samples (i.e. doubling the total number of positive training samples to 8032), to better represent the head to be detected. These positive training samples were cropped from the training images; the negative training samples were obtained from a combination of our crowd

scenes, internet images and Microsoft Clip Art Gallery. The last contains many cartoon images, which have mostly solid colors, and account for the "spikes" in the normalized average RGB histograms in Figures 4.5(b) and 4.5(d). The training process took a few days on a single core 3.4GHz PC with 3GB memory.

We also trained a Dalal and Triggs head detector as in [70], which uses HOG descriptors instead of Haar-like features. The implementation was obtained from the INRIA Object Detection and Localization Toolkit [77] where the cell and block sizes were tuned to 6 x 6 and 3 x 3 pixels, respectively, to suit the scenario here. The positive and negative training samples used here correspond to the edge maps of the samples used to train the Viola-Jones head detector. The training process took less than a day on a single core 3.4GHz PC with 3GB memory.

4.3.2 Results and Discussion

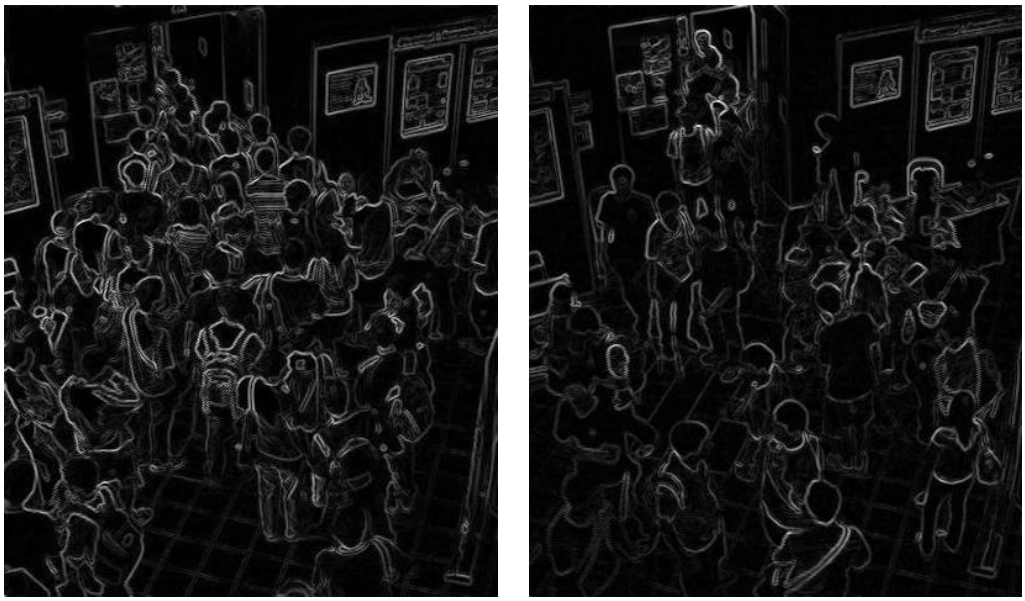
A test set, different from the training images, with a total of 30 still images of dense crowds with an average density of 35 persons per image was used to evaluate both the head detectors. The ground truth for true positive, false positive and true negative detections of a head was manually marked. The Viola-Jones head detector yielded a detection accuracy of 87.8% and false alarm rate of 35.7% (e.g. Figure 4.4); whereas the Dalal and Triggs head detector yielded a detection accuracy of 66.0% and false alarm rate of 38.9% (e.g. Figure 4.12). The possible reason for the poor performance using HOG descriptors is the loss of low frequency components in the edge maps used in the Dalal and Triggs approach, unlike Viola-Jones approach where the multi-scale processing better preserves all the frequencies. From the two edge maps in Figure 4.13 where the low frequency components are suppressed, it is nearly impossible to locally detect some heads even by visual observation. [78] which uses the same Viola-Jones approach, reported better results than we have obtained, but their data consisted only

of frontal faces. These results are reasonable because the dense crowd scenario includes occlusions and different orientations of the individuals. Also, the work reported in [30] was able to segment out individuals on cluttered backgrounds in still images regardless of their orientations, but the reported detection rate of 82.5% at a false alarm rate of 32.5% was obtained on less dense crowds of at most 10 people per image (e.g. Figure 2.2(a)). In summary, the Viola-Jones detector performs better in the scenario of interest, and hence we have chosen it to provide the initial head detections.

(a) (b)

Figure 4.12: Example images of dense crowds showing detections with Dalal and Triggs head detector: (a) Detections on Figure 1.3(c); (b) Detections on Figure 1.3(d).

The detected windows from this initial pass were further classified with the color bin image approach. Comparison of Figures 4.4 and 4.9 shows that a few true detections have been misclassified by the color bin image classifier, but a large number of false alarms have been removed; this is especially evident from Figure 4.4(b) and Figure 4.9(b). The detection rate for the same 1053 people in the 30 test images is

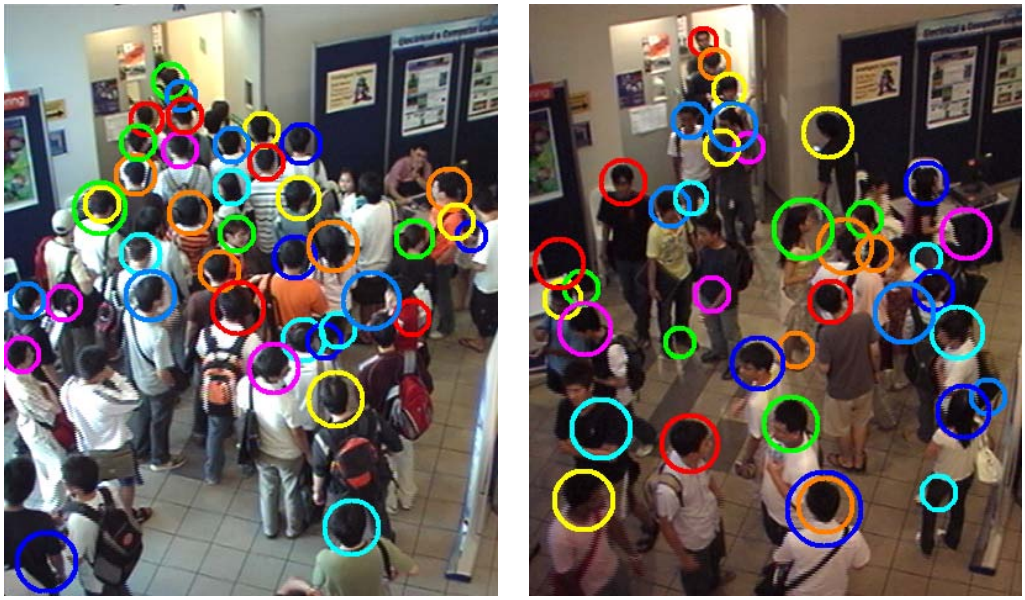


(a) (b)

Figure 4.13: Example edge maps of dense crowds: (a) Edge map of Figure 1.3(c); (b) Edge map of Figure 1.3(d).

slightly reduced to 84.8% but the false alarm rate is greatly reduced to 26.0%, compared to the detection rate of 87.8% and false alarm rate of 35.7% in the first pass. Finally, when the regression line approach was used on the windows obtained from the color bin image classifier, the final results obtained are as in Figure 4.14. From the figure, it is obvious that the biggest and the smallest circles corresponding to the false alarms in Figure 4.9 have been removed effectively. The final detection rate was 79.0% with a false alarm rate of only 20.3%.

We tested the performance for the same set of images at different tolerance levels to obtain the ROC curves shown in Figure 4.15. It can be seen from the figure that our color bin image approach significantly improves the performance of the initial Viola-Jones head detector. For example, at a detection rate of 84.0%, the false alarm rate of the initial detector is 30.6%, whereas if used with the color bin image approach, the false



(a) (b)

Figure 4.14: Example images of dense crowds showing final detections after color bin image and regression line approaches: (a) Resultant image where Figure 1.3(c) is the input image; (b) Resultant image for Figure 1.3(d).

alarm rate drops below 26.0%. By combining the color bin image and regression line approaches, the ROC curve in Figure 4.15 shows improvement of false alarm rates to about 24.0%. The detection accuracy is about 80.0% at this false alarm rate. However, at false alarm rates above 24.0%, the accuracy of the combined detector is lower than when using only the color bin image approach. This is because the regression line approach uses a least squares line fitting method, which will be affected by a large number of false alarms (outliers). The ROC curves indicate the relative superiority of the different methods we proposed depending on the false alarm and detection accuracy trade-off that will be acceptable in a given application. For false alarm rates between 15.0% - 24.0%, the combined detector with color bin image and regression line fitting is the best, offering detection accuracy in the 70.0% - 81.0% range. For false alarm rates between 25.0% - 42.0%, using only the color bin image approach is superior, offering detection rates between 81.0% - 91.0%.

For an informal comparison with the ROC curve of Seemann et al. [30] with ours, we have plotted it in Figure 4.15 (the crowd scenes considered in [30] and here are different). It can be seen that our detection rates are significantly higher. Moreover, our detectors can yield accuracies above 90.0% detection rate, while the method of [30] saturates below 90% accuracy, even though our crowd density is more than three times that of [30] which has less than 10 people per image.

4.3.3 Experimenting on different scenes

To test the robustness of our approach, we further experimented on two different crowded scenes where the assumption of an indoor environment with a camera mounted at an elevated level, does not hold. Figure 4.16 shows two different scenes of dense crowds: the first one is taken with the camera at a lower level, instead of being elevated; the second one shows an outdoor dense crowd instead of an indoor dense crowd.

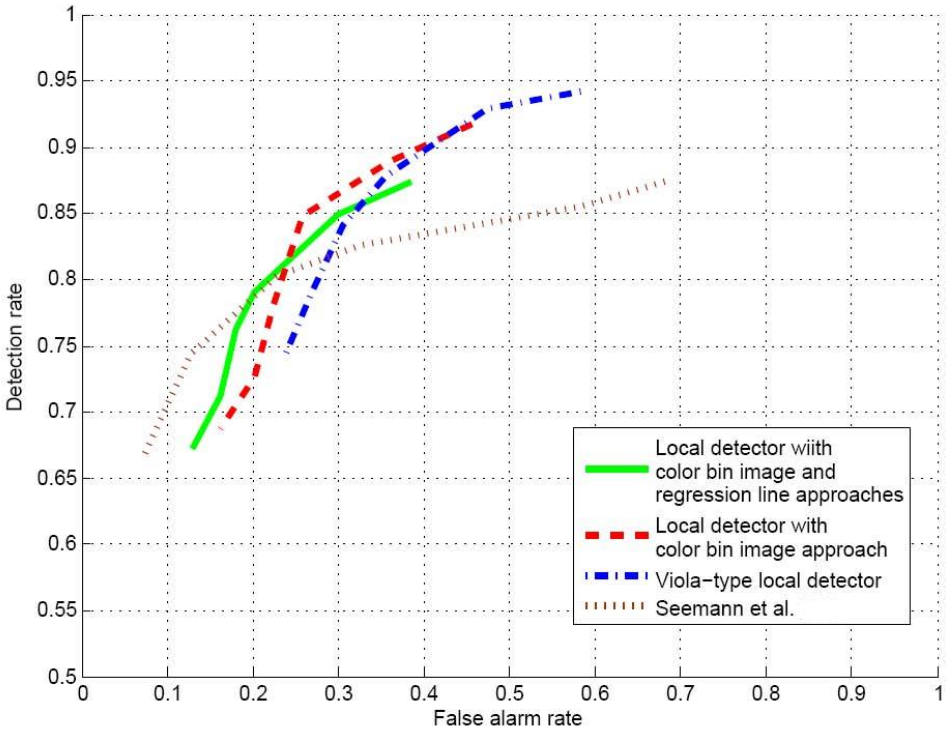


Figure 4.15: Receiver Operating Characteristics curves.

Figure 4.17 shows the detection results on these two scenes. Even though these scenes are different from the ones used for training and the camera viewpoint is different, our head detection approach is still able to obtain reasonable results. However, the detector performance on these scenes is not as good as that on the original test scenes, which is to be expected (for example, Figures 4.17(a) and 4.17(b) have more false alarms than Figures 4.14(a) and 4.14(b)).

(a) (b)

Figure 4.16: Different scenes of dense crowds from [79]: (a) Camera shooting from a lower level view; (b) An outdoor dense crowd.

(a) (b)

Figure 4.17: Detection results on different scenes of dense crowds from [79]: (a) Detection results for Figure 4.16(a); (b) Detection results for Figure 4.16(b).

4.4 Summary

We experimentally compared the Viola-Jones detector and the Dalal and Triggs detector for local head detection in densely crowded scenes, and found that the former performed better in this scenario; hence, we use it in our work as the initial head detector. To reduce false alarms, we proposed an approach using color bin images and supervised learning of a cascade of boosted classifiers using Haar-like features. The color bin images are formed from the initially detected windows. We also proposed another method which uses weak perspective modeling with a single uncalibrated camera for further reducing false alarms. Unlike other works, this only relies on the 2D image size and the 2D locations of the detections in images and does not require any 3D world information. Our only assumptions are that the people in the scene have the same 3D world size and the crowd distributes over a plane. An approximately linear relationship is found between image size and vertical location in the image. An algorithm based on least squares line fitting is then used to remove detections that are considered as false alarms (outliers).

Results of our proposed approaches show significant improvements compared to the Viola-Jones-type head detector: the initial head detector coupled with the color bin image approach or the detector coupled with color bin image and regression line approaches give promising results for detecting individuals within dense crowds. We further experimented with crowded scenes that differed from the training scenes, and found that the detector yielded reasonably good performance. However, our approach is color-based and its performance may be challenged by the color intensities of the heads to be detected. Hence, for optimum performance, the classifiers will need to be tuned to the scenes of interest.

Chapter 5

Tracking Individuals in Dense Crowds

As mentioned in Section 1.1.2, a tracking system can be implemented jointly with a detection system or as an independent module that only uses a detection system for initialization. This chapter focuses on the latter type, to facilitate real-time video tracking; the former is computationally more intensive, and real-time video tracking may not be possible. However, an independent tracking module faces the problem of finding the objects' location in every frame, since object detections are only given to the tracking system in the first frame and the tracker has to predict and update the objects' locations in subsequent frames based on data from previous and present frames. In this chapter, we first review Bayesian filtering, the optimal tracking technique which combines prediction and data-updating, and then propose a tracking system for individuals in dense crowds. Next, we present the experimental results and the conclusion.

5.1 Bayesian Filtering

In object tracking, Bayesian filtering estimates the state (location, region, etc.) of the object from noisy measurements or observations. It is an optimal technique which uses all available information for tracking. In the frame at time t, the filter computes the posterior density of the object state conditioned on information from present and past frames, given as

$$Bayes(x_t) = p(x_t|z_0, z_1, ..., z_t) \tag{5.1}$$

where x_t and z_t are the state and observation(s), respectively, at time t.

From Bayes Theorem, $Bayes(x_t)$ can be computed from three other densities: the prior, $p(x_t|z_0, z_1, ..., z_{t-1})$; the likelihood, $p(z_t|x_t)$; and the evidence, $p(z_t|z_0, z_1, ..., z_{t-1})$. This computation can be done recursively by writing

$$Bayes(x_t) = \frac{p(x_t|z_0, z_1, ..., z_{t-1})p(z_t|x_t)}{p(z_t|z_0, z_1, ..., z_{t-1})} \tag{5.2}$$

Assuming that the state process is first order Markov, the prior can be written as

$$\begin{aligned} p(x_t|z_0, z_1, ..., z_{t-1}) &= \int p(x_t|x_{t-1})p(x_{t-1}|z_0, z_1, ..., z_{t-1})dx_{t-1} \\ &= \int p(x_t|x_{t-1})Bayes(x_{t-1})dx_{t-1} \end{aligned} \tag{5.3}$$

and the evidence density as

$$p(z_t|z_0, z_1, ..., z_{t-1}) = \int p(x_t|z_0, z_1, ..., z_{t-1})p(z_t|x_t)dx_t \tag{5.4}$$

which serves as a normalizing factor. This yields the recursive estimate for $Bayes(x_t)$ as

$$Bayes(x_t) = \frac{p(z_t|x_t) \int p(x_t|x_{t-1}) Bayes(x_{t-1}) dx_{t-1}}{\int \int p(x_t|x_{t-1}) p(z_t|x_t) Bayes(x_{t-1}) dx_t dx_{t-1}} \tag{5.5}$$

and the optimal state estimate is obtained as

$$\widehat{x}_t = \max_{x_t} Bayes(x_t) \tag{5.6}$$

The only initialization to be made is for $Bayes(x_0)$, which is usually assumed uniformly distributed if no prior knowledge of the object's location is available. For more details on Bayesian filtering, refer to [80].

In summary, Bayesian filtering provides a probabilistic framework for recursive state estimation, using the data-update component (or likelihood) $p(z_t|x_t)$, and the prediction component $p(x_t|x_{t-1})$, to construct $Bayes(x_t)$. The process is depicted in Figure 5.1. If the object dynamics are unknown, $p(x_t|x_{t-1})$ can simply be assumed to be a uniform density, in which case Equation 5.5 reduces to a maximum likelihood estimate,

$$x_t^{max} = \max_{x_t} \frac{p(z_t|x_t)}{\int p(z_t|x_t) dx_t} \tag{5.7}$$

which uses only the data-update component.

In the following we further elaborate on the prediction and data-update components and give examples of tracking systems that focus on different components.

5.1.1 Prediction Component

Prediction uses $p(x_t|x_{t-1})$ to model the dynamics of the tracked object. The two most popular Bayesian tracking approaches that use the prediction component are Kalman filtering [81, 82] and particle filtering [83, 84]. The Kalman filter predicts only a single new state for each object tracked, and updates the state based on the error

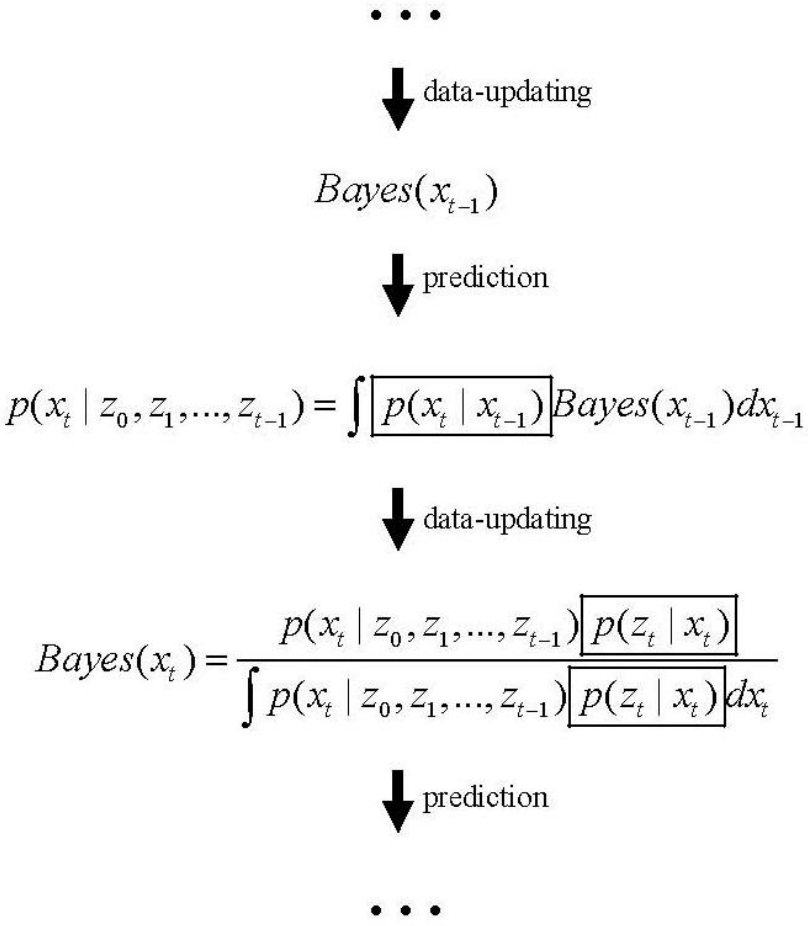


Figure 5.1: Bayesian filtering process

between the predicted state and the observed data. Despite the low computation cost for Kalman filtering, it is optimal only for uni-modal or Gaussian densities. Further, the basic Kalman filter is linear, and to work on nonlinear problems, specific linearization steps are needed to obtain the extended Kalman filter. Whereas, particle filtering is suitable for both linear or nonlinear dynamical systems with multi-modal distributions. Moreover, particle filtering also predicts multiple possible states for each object tracked, whose probabilities are estimated based on the current observed data and the assumed probability distributions to obtain the final state. Being able to predict multiple states, particle filtering is more robust with respect to prediction error, but it is also more highly computationally intensive, when compared to Kalman filtering.

Tracking systems that use prediction perform effectively if the dynamics of the objects can be accurately modeled, e.g. tracking flying birds [85], tracking heads and hands in simple environments [86], and tracking vehicles in aerial video surveillance [87]. The obvious advantage of these trackers is their ability to estimate the objects' locations under temporary occlusions, since they compute the posterior densities based not only on observations, but also on the prediction component.

5.1.2 Data-updating Component

In contrast to the prediction component, the data-updating component which is solely based on observations allows the system to cope with changes in object appearance. Tracking approaches that use maximum likelihood estimation require only the data-updating component $p(z_t|x_t)$, which can be specified in different ways. Examples are object appearance models (which are learned offline), including 3D models, blob models, adaptive mixture models and even motion models (see Section 2.1). On the other hand, there are other maximum likelihood approaches that are less problem-specific, e.g. mean shift tracking [21] which estimates and uses regional feature (i.e. color, tex-

ture or edges) histograms of objects; active contour models [22] which use object edges for tracking; and KLT [53] which uses strongly-textured feature points of objects. Comparatively, the problem-specific tracking approaches give better performance in heavily cluttered scenes but are computationally more complex. These tracking approaches can be incorporated into Bayesian tracking schemes, or they can be used as standalone trackers in a maximum likelihood framework (which is equivalent to assuming a uniformly distributed prediction density function).

Tracking systems that rely mainly on data-updating, are more suited for tracking in crowded or cluttered scenes where the appearance of the object plays a more vital role than the object dynamics, e.g. face tracking within dense crowds [18], human tracking in crowded environments [31] and vehicle tracking against cluttered background [35]. However, tracking systems that strictly depend on the data-updating component lack predictive ability and cannot handle occlusion problems.

5.2 Proposed Tracking System: A Bayesian Approach

Here, we propose a tracking scheme that uses an object detector only for initialization, and takes over the tracking thereafter; this facilitates real-time video tracking. More specifically, the primary aim of the proposed scheme is to track individuals mingling within a dense crowd, given the detections from the detector of Chapter 4 as initializations. However, the proposed scheme can also track other objects in different scenarios, as long as the initial object detection window is given. As discussed in Chapter 2, for a dense crowd as shown in Figure 1.3, none of the currently available tracking approaches is suitable.

In the Bayesian approach, the posterior density of the object state is calculated conditioned on observations from present and past frames (Equation 5.1). In our proposed tracking scheme, we assume that the observations obtained from a pair

of consecutive frames at each time instant is sufficient to estimate the object state, simplifying Equation 5.1 to

$$Bayes(x_t) = p(x_t|z_t) \tag{5.8}$$

and the optimal state estimate from Equation 5.6 becomes

$$\widehat{x}_t = \max_{x_t} p(x_t|z_t) \tag{5.9}$$

Closest to our proposed tracker is the mean shift tracker, in that both seek to track an initialized object without being dependent on any specific object type. However, mean shift tracking uses a maximum likelihood approach which relies only on spatial information, while our tracker uses a Bayesian approach which includes temporal measurements. In brief, our tracker uses several KLT feature points on the object and computes their weights (relative importance) for Bayesian tracking. However, during time instants when object occlusion is inferred, the tracker uses a simple linear approximation to estimate the object's location. Also, additional characteristics of the tracker, such as robustness to scaling and rotational motions, are proposed.

5.2.1 Bayesian tracker: Motion Coherence of Feature Points as Observations

As in Equations 5.8 and 5.9, we first aim to estimate the posterior density, $p(x_t|z_t)$, over all available object states x_t, for finding $\widehat{x}_t$, the optimal object state which maximizes $p(x_t|z_t)$. Similar to the mean shift tracker (and others), we propose to track the object of interest by estimating in each frame, the center of gravity (CG) of the image window enclosing the object. Also, as in the mean shift tracker, we assume that the image window is initialized in the first frame such that the box enclosing the object of interest

is correct (i.e. object is within the image window, be it partially occluded or not. See Figure 5.2).

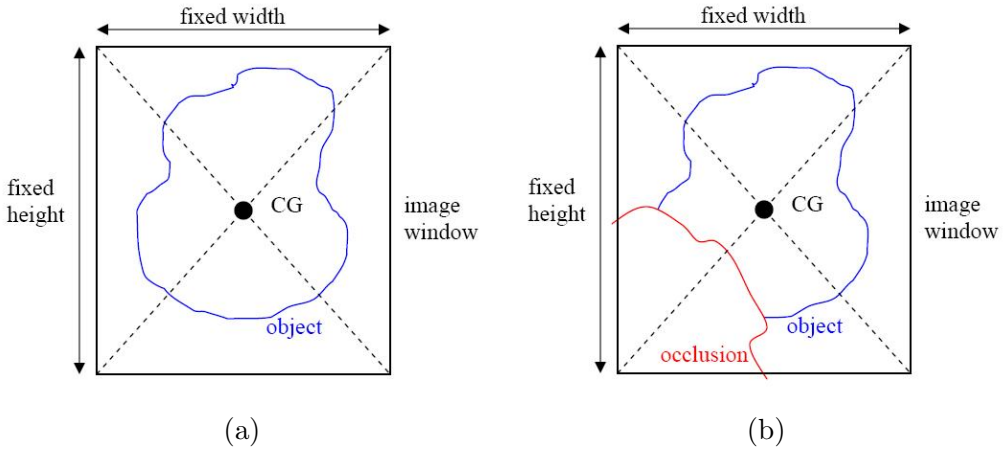


Figure 5.2: Examples of image window on object of interest: (a) Image window on object; (b) Image window on partially occluded object.

We first define x_t as the estimated position vector of the image window's CG at time t. Next, we define f_i as the i^{th} feature point and $\mathbf{F} = \{f_i, i = 1, 2, ..., N\}$ as the set of all N feature points, which are detected within the image window whose CG was estimated as $\widehat{x}_{t-1}$ in the previous frame. Assuming an ideal case where all N feature points are tracked to the current frame at time t, the observations z_t, are defined to be the motion vectors of the N feature points. In the context of this subsection, we denote $\overrightarrow{m_i}$ and $\overrightarrow{M}$ as the motion vectors of f_i and the image window's CG, respectively, from the previous frame to the current (i.e. $\overrightarrow{M}$ is an unknown that is to be estimated), as shown in Figure 5.3 for $N = 8$. From this, our goal is to find $\widehat{x}_t$, the position vector of the image window's CG at time t that corresponds to the maximum of $p(x_t|z_t)$, by estimating $\overrightarrow{M}$ such that $\widehat{x}_t = \widehat{x}_{t-1} + \overrightarrow{M}$.

To find $\widehat{x}_t$, note that $p(x_t|z_t)$ is an N-discrete distribution, where z_t are the motion observations for the N feature points at time t. x_t^i denotes the N position

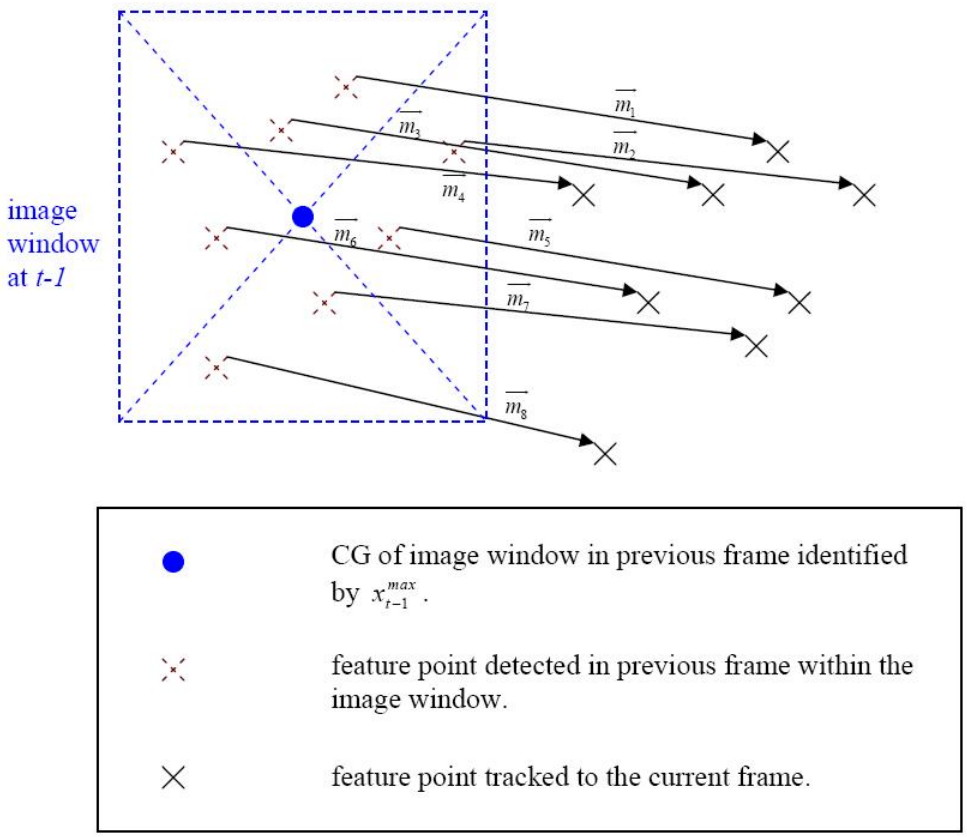


Figure 5.3: The available observations (strongest KLT features in an image window at $t-1$). In this ideal case, all $N = 8$ features are tracked to the current frame at time t.

vectors formed from the sum of $\widehat{x}_{t-1}$ and $\overrightarrow{m_i}$, i.e. $\widehat{x}_{t-1} + \overrightarrow{m_i}, i = 1, 2, ..., N$. Each x_t^i has a probability conditioned on the observations at time t (i.e. $P(x_t^i = \widehat{x}_{t-1} + \overrightarrow{m_i}|z_t)$), and the optimal estimate of the image window's CG at time t is the mode of $p(x_t|z_t)$, i.e. $\widehat{x}_t = \widehat{x}_{t-1} + \overrightarrow{m_o}$, where $o = \arg\max_{i=1,2,...,N} P(x_t^i = \widehat{x}_{t-1} + \overrightarrow{m_i}|z_t)$, where $\overrightarrow{m_o}$ is the optimal estimate for $\overrightarrow{M}$. The rest of this subsection describes how $P(x_t^i = \widehat{x}_{t-1} + \overrightarrow{m_i}|z_t)$, denoted as P_i, is formulated and used to track the object by estimating the image window's CG in every frame.

Formulating P_i

As above, $\widehat{x}_t = \widehat{x}_{t-1} + \overrightarrow{m_i}$ implies that $\overrightarrow{M} = \overrightarrow{m_i}$ and z_t denotes the motion vectors of the feature points. Here, we propose to compute P_i by defining it as the conditional probability that $\overrightarrow{M} = \overrightarrow{m_i}$ given the motion coherence (defined below) between the image window's CG and all other feature points. We first assume only translational object motion (this restriction is subsequently removed), and hence we estimate the object's location using an image window of fixed height and width. As shown in Figure 5.2, given an image window as the output of an object detector at time $t = 0$, or an image window estimated from the tracking system at time $t - 1$, $t > 1$, we first detect the N strongest-textured KLT feature points within the image window and track them, from time $t - 1$ to time t (refer to Section 3.1 for KLT tracking). Feature points that are not tracked by the KLT tracker (due to sudden occlusion or motion blur), are deleted from the set $\mathbf{F}$ since motion vectors of these feature points cannot be estimated. As suggested in Figure 5.2, the majority if not all of the KLT feature points can usually be expected to lie on the object, unless the non-object regions within the window, such as the background or an occluding object, have higher textures (this will be dealt with in the definition of the prior probability below). If an object motion is such that its net translational motion is greater than its articulatory movements (if any) and if the

majority or all of the tracked KLT feature points are on the tracked object, the more the motion vectors of the other feature points are similar to a given $\overrightarrow{m_i}$, the more likely it is that this vector is a suitable estimate for $\overrightarrow{M}$. Hence, the similarity of $\overrightarrow{m_i}$ with the motion vectors of other feature points is a reasonable measure to compute P_i. We term this similarity as the motion coherence between f_i and the other feature points. A Bayesian scheme for computing P_i is proposed as follows:

$$\begin{aligned} P_i &= P(\overrightarrow{M} = \overrightarrow{m_i}|B, C) \\ &= \frac{P(\overrightarrow{M} = \overrightarrow{m_i}, B, C)}{P(B, C)} \\ &= \frac{P(\overrightarrow{M} = \overrightarrow{m_i})P(B|\overrightarrow{M} = \overrightarrow{m_i})P(C|\overrightarrow{M} = \overrightarrow{m_i})}{P(B)P(C)} \end{aligned} \tag{5.10}$$

where B and C refer to the motion coherence between the image window's CG and all other feature points in $\mathbf{F}$, based on motion vector magnitudes, and phase, respectively. It is reasonable to assume in the above that B and C are independent. Below we define the prior and conditional probabilities in Equation 5.10.

Definition of Prior Probability

The motion vector $\overrightarrow{m_i}$ of f_i is defined based on $\overrightarrow{O_i}(t-1)$ and $\overrightarrow{O_i}(t)$, the position vectors of f_i at time $t-1$ and time t, respectively (i.e. $\overrightarrow{O_i}(t) - \overrightarrow{O_i}(t-1) = \overrightarrow{m_i}$). To handle the problematic situation when non-object regions within the windows have higher textures, and lead to feature points being concentrated there, we need to ensure that the spatial distribution of the feature points within the image window is as uniform as possible, so that a sufficient number of feature points will also be detected on the object. Firstly, this can be done by finding K features ($K >> N$) and selecting N of these which are strong enough and are as uniformly distributed as possible in the window, though this may

mean that these feature points may not be the top N strongest-textured KLT feature points within the image window. However, this is the trade-off necessary to ensure that a sufficient number of feature points lie on the object. Secondly, an isotropic measure centered at $\widehat{x}_{t-1}$ is used to compute $P(\overrightarrow{M} = \overrightarrow{m_i})$, the prior probability, such that feature points which are closer to the CG have higher probability of having the same motion vector as $\overrightarrow{M}$ than those farther away from the CG. This is given by Equation 5.11 below where β is a normalizing factor for $P(\overrightarrow{M} = \overrightarrow{m_i})$ to be a probability distribution. The decrease in $P(\overrightarrow{M} = \overrightarrow{m_i})$ as the feature point moves farther from the CG is a reasonable way to discount feature points on non-object regions because feature points that are closer to the CG are more likely to be on the object since only partial occlusion cases are considered here (refer to Figure 5.2).

$$P(\overrightarrow{M} = \overrightarrow{m_i}) = \frac{\beta}{1 + ||\widehat{x}_{t-1} - \overrightarrow{O_i}(t-1)||^2} \tag{5.11}$$

Definition of Likelihood

In Equation 5.10, $P(B|\overrightarrow{M} = \overrightarrow{m_i})$ is the likelihood of coherent motion between the image window's CG and all other feature points in $\mathbf{F}$, given that $\overrightarrow{M} = \overrightarrow{m_i}$, based on the magnitude of the motion vectors. Likewise for $P(C|\overrightarrow{M} = \overrightarrow{m_i})$ with respect to motion vector phase. Since it is reasonable to measure the motion coherence based on the length and direction of motion vectors, we define $P(B|\overrightarrow{M} = \overrightarrow{m_i})$ as in Equation 5.12 where γ is a normalizing factor for $P(B|\overrightarrow{M} = \overrightarrow{m_i})$ to be a probability. The idea is that the closer the magnitude of $\overrightarrow{m_i}$ is to all other motion vectors, the more likely that they are in coherent motion. Likewise, $P(C|\overrightarrow{M} = \overrightarrow{m_i})$ is defined in Equation 5.13. Again, we represent these likelihoods as reciprocal functions to suppress motion vectors that are not similar to all other motion vectors, because they are likely to be from non-object

regions.

$$P(B|\overrightarrow{M} = \overrightarrow{m_i}) = \gamma \prod_{i \neq j} \frac{1}{1 + (||\overrightarrow{m_j}||_2 - ||\overrightarrow{M}||_2)^2} \quad , \quad ||\overrightarrow{M}||_2 = ||\overrightarrow{m_i}||_2 \tag{5.12}$$

$$P(C|\overrightarrow{M} = \overrightarrow{m_i}) = \zeta \prod_{i \neq j} \frac{1}{1 + (\angle\overrightarrow{m_j} - \angle\overrightarrow{M})^2} \quad , \quad \angle\overrightarrow{M} = \angle\overrightarrow{m_i} \tag{5.13}$$

Definition of Evidence Factor

The evidence factor is a normalization constant in the denominator of Equation 5.10 which ensures that the posterior probability sums up to one, and is defined as:

$$\begin{aligned} P(B)P(C) &= P(B,C) \\ &= \sum_i P(B,C|\overrightarrow{M} = \overrightarrow{m_i})P(\overrightarrow{M} = \overrightarrow{m_i}) \\ &= \sum_i P(B|\overrightarrow{M} = \overrightarrow{m_i})P(C|\overrightarrow{M} = \overrightarrow{m_i})P(\overrightarrow{M} = \overrightarrow{m_i}) \end{aligned} \tag{5.14}$$

After P_i is computed for all f_i in $\mathbf{F}$, we obtain $\overrightarrow{M} = \overrightarrow{m_o}$, $o = \arg\max_{i=1,2,\ldots,|\mathbf{F}|} P_i$ for the motion vector of the CG. The estimated object's CG at time t is then obtained as $\widehat{x}_t = \widehat{x}_{t-1} + \overrightarrow{M}$. This process is repeated in every frame of a video sequence as long as no significant object occlusion is inferred. Next, we present a way to estimate the CG by a linear approximator when the object is partially occluded.

5.2.2 Linear Approximation: Estimating the Mean Velocity

The tracker normally uses the Bayesian framework above for estimating $\widehat{x}_t$, the image window's CG. However, if the tracker infers that more than half of the object is occluded, it will switch to using linear approximation to estimate $\widehat{x}_t$, as shown in Figure 5.4. Occlusion is inferred if the KLT tracker fails to track more than $\frac{N}{2}$ feature points from the reference frame at $t-1$ to the current frame at t. The Bayesian framework will only take over from the linear approximator when at least $\frac{N}{2}$ feature points of the reference frame, just before occlusion takes place, are tracked by the KLT tracker in the linearly approximated image window of the current frame.

We use a $(q-1)$th order linear approximator for $\widehat{x}_t$ based on the q CG locations $\widehat{x}_{t-1}, \widehat{x}_{t-2}, \widehat{x}_{t-3}, ..., \widehat{x}_{t-q}$. With the motion vector of the CG defined as $\overrightarrow{M}(t-1) = \widehat{x}_{t-1} - \widehat{x}_{t-2}$, we estimate $\widehat{x}_t$ as:

$$\widehat{x}_t = \widehat{x}_{t-1} + \frac{1}{q-1} \sum_{i=1}^{q-1} \overrightarrow{M}(t-i) \tag{5.15}$$

where $\frac{1}{q-1} \sum_{i=1}^{q-1} \overrightarrow{M}(t-i)$ is the mean velocity over q frames.

5.2.3 Building Robustness to Scaling and Rotation

The development thus far has considered only translational motion, but scaling and rotational motions also occur in real-world scenarios and must be taken into consideration. Scaling occurs when object distance to camera changes, and the fixed image windows as used previously may become too small or too large for the object. If rotational motion is treated simply as translational motion, the image window can be displaced from the object. The rest of this subsection proposes techniques to deal with scaling and rotational motions.

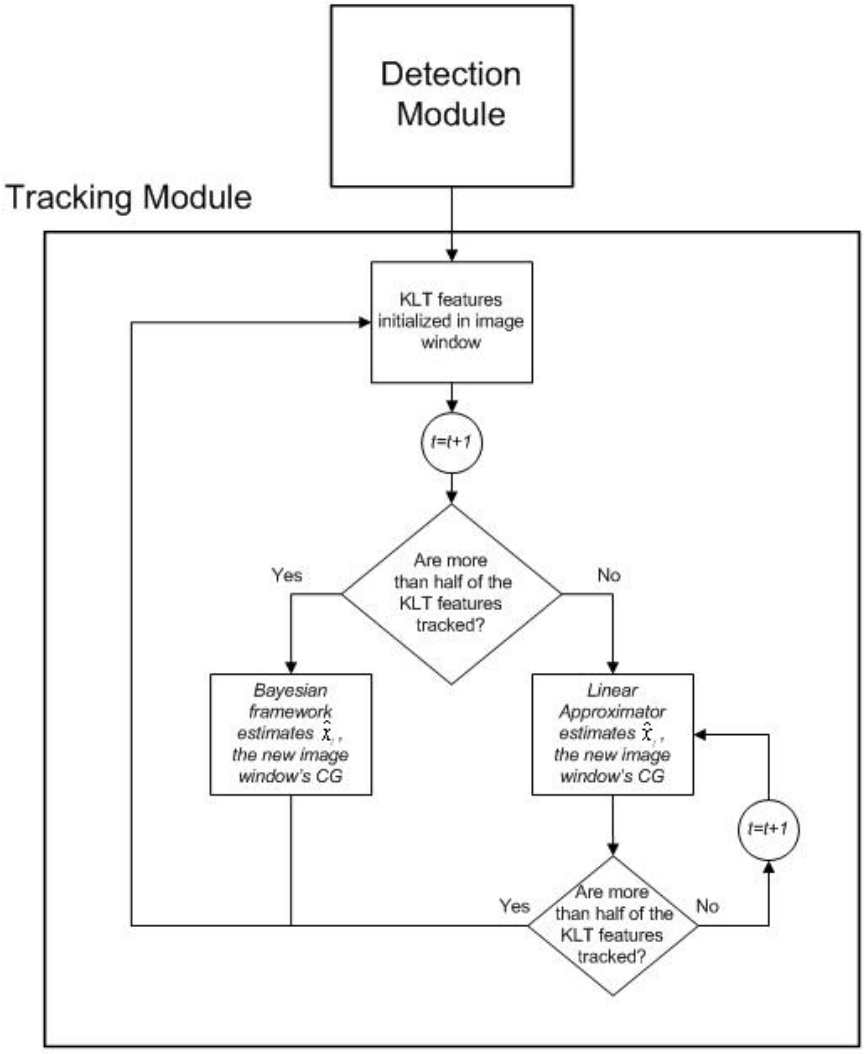


Figure 5.4: The proposed complete tracking system.

Resizing Image Window under Scaling Motion

Figure 5.5 shows how an image window should ideally resize with the scaling of an object. For rigid body scaling, the distance between any two feature points is linearly related to the width or height of the image window. However, as we assume non-rigid object motion where the net translational motion is greater than the articulatory movement, the mean distance between all feature points is a better estimate for the linear relationship with the image window size. Also to discount for feature points on non-object regions as in Figure 5.5, we give less weight to feature points that are further away from the image window's CG (as we did in the definition of $P(\overrightarrow{M} = \overrightarrow{m_i})$, where $\overrightarrow{m_i}$ refer to the motion vectors from $t-1$ to t). Using these ideas, we propose to determine the new height, H_t, and new width, W_t, of the image window from the previous height, H_{t-1}, and width, W_{t-1}, as:

$$H_t = \lambda_t H_{t-1} \tag{5.16}$$

$$W_t = \lambda_t W_{t-1} \tag{5.17}$$

where $\lambda_t = \frac{\sum_{i,j} (P(\overrightarrow{M}=\overrightarrow{m_i})+P(\overrightarrow{M}=\overrightarrow{m_j}))\ ||\overrightarrow{O_i}(t)-\overrightarrow{O_j}(t)||_2}{\sum_{i,j} (P(\overrightarrow{M}=\overrightarrow{m_i})+P(\overrightarrow{M}=\overrightarrow{m_j}))\ ||\overrightarrow{O_i}(t-1)-\overrightarrow{O_j}(t-1)||_2}$

Handling Rotational Motion in Tracking

Generally, there are two different approaches to tracking objects undergoing rotational motion: using 3D knowledge or different views of the object known a priori (as in [88]), or using model-based approaches as described in Section 2.1. Another category tracks the contours or shapes of the objects using contour based approaches as described in Section 2.1, and bypasses the tracking challenges arising from rotational motion. However, as our tracker is feature-based, we can only make use of the observations

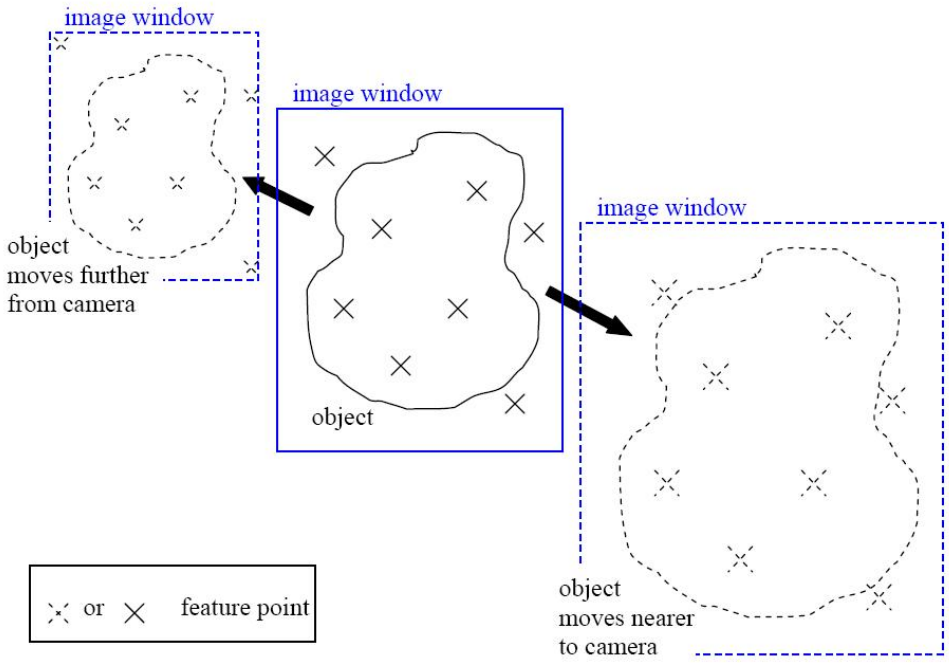


Figure 5.5: Ideal scaling of image window together with the object.

from the KLT feature points to infer an object's rotational motion. Since upwards and downwards rotational motion of an object are rare and do not pose much problems in surveillance videos, we only consider panning rotational motion, which is usually confused with a leftwards or rightwards translational motion when using feature-based approaches. For example, the image window of the object at time $t-1$ may be correctly estimated at first, but because the object is rotating leftwards, the image window at time t is expected to be displaced as shown in Figure 5.6. [89] tackles this problem by assuming the feature points to be newly revealed parts of the object ('f' in Figure 5.6) if they moved consistently with the known parts of the object ('b' to 'e' in Figure 5.6). Applying the same assumption, the following describes our approach for dealing with this common problem in feature-based approaches.

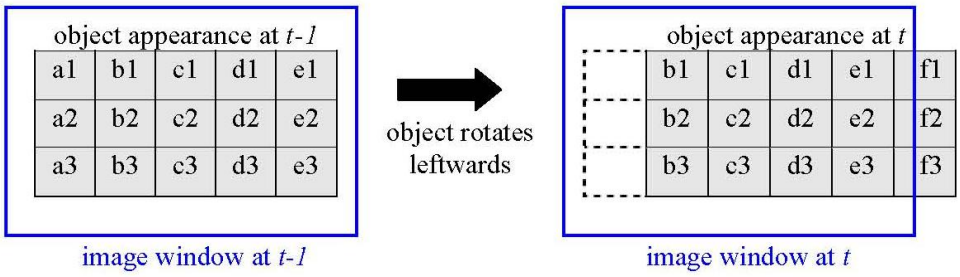


Figure 5.6: The common problem of image window displacement, faced by feature-based approaches when the object rotates left (pans), where the the object and non-object regions are denoted by grey and white regions respectively.

First, we split the image window at time $t-1$ into left and right halves. Then for each half, the motion coherence measure between every feature point and all other feature points in $\mathbf{F}$ (equivalent to the likelihood term of P_i), is calculated from time $t-1$ to t. The means of the motion coherence measures for the left and the right halves

are denoted as μ_L and μ_R, respectively, and defined in Equations 5.18 and 5.19 as

$$\mu_L = \frac{\sum_i P(B|\overrightarrow{M} = \overrightarrow{m_i})P(C|\overrightarrow{M} = \overrightarrow{m_i})}{|\mathbf{m_L}|} \quad , \quad \forall \overrightarrow{m_i} \in \mathbf{m_L} \tag{5.18}$$

$$\mu_R = \frac{\sum_i P(B|\overrightarrow{M} = \overrightarrow{m_i})P(C|\overrightarrow{M} = \overrightarrow{m_i})}{|\mathbf{m_R}|} \quad , \quad \forall \overrightarrow{m_i} \in \mathbf{m_R} \tag{5.19}$$

where $\mathbf{m_L}(\mathbf{m_R})$ is the set of motion vectors from the left(right) half of the image window and $|\mathbf{m_L}|(|\mathbf{m_R}|)$ refers to the number of motion vectors from the left(right) half of the image window.

Next, we resolve $\overrightarrow{M}$, the motion vector of the image window's CG, estimated from our Bayesian framework at time t, into its vertical and horizontal components (i.e. $\overrightarrow{M} = M_v\hat{v} + M_u\hat{u}$, where $\hat{v}$ and $\hat{u}$ are unit vectors pointing in the upward and rightward directions in the image, respectively).

Following [89], and since the feature points within the image window are chosen to be uniformly distributed, for the case when M_u is negative (i.e. the estimated $\widehat{x}_t$ is to the left of $\widehat{x}_{t-1}$), if $\frac{\mu_L}{\mu_R} \approx 1$, a leftwards translational motion of the object is implied, while if $\frac{\mu_L}{\mu_R} < 1$, a leftwards rotational motion is implied. These situations are illustrated in Figures 5.7(a) and 5.7(b), from where we can see that if $\frac{\mu_L}{\mu_R} \approx 1$, the estimated $\overrightarrow{M}$ from the Bayesian framework, which assumes only translational motion, should be used for approximating the object's motion; if $\frac{\mu_L}{\mu_R} < 1$ because of leftwards rotational motion, the magnitude $|M_u|$ should be made smaller for the image window to enclose more of the object. Likewise for rightwards rotational motion, where $\frac{\mu_L}{\mu_R} > 1$. Hence, we simply use $\frac{\mu_L}{\mu_R}$, bounded by a threshold τ to prevent overestimating M_u, as a linear factor for the magnitude of M_u.

Also, two other possibilities when the Bayesian framework estimates $\widehat{x}_t$ to the left of $\widehat{x}_{t-1}$, are shown in Figures 5.7(c) and 5.7(d): a displacement of the image window

Algorithm 2 Algorithm for handling rotational motion.

for each $\overrightarrow{M}$ estimated (at time t) **do**

if $M_u < 0$ **then**

if $\frac{\mu_L}{\mu_R} < \tau$ **then**
$M_u = \frac{\mu_L}{\mu_R} \times M_u$;
else
$M_u = \tau \times M_u$;
end if

else

if $\frac{\mu_R}{\mu_L} < \tau$ **then**
$M_u = \frac{\mu_R}{\mu_L} \times M_u$;
else
$M_u = \tau \times M_u$;
end if

end if

end for

to the left and a displacement of the image window to the right, respectively. These cases are common when a feature-based tracker makes feature tracking errors after some frames, due to poor matching. We can further observe from Figures 5.7(c) and 5.7(d) that if $\frac{\mu_L}{\mu_R} < 1$, the magnitude $|M_u|$ should be made smaller for the image window to enclose more of the object and if $\frac{\mu_L}{\mu_R} > 1$, the magnitude $|M_u|$ should be made bigger for the image window to enclose more of the object. Hence, besides handling rotational motions, Algorithm 2 likewise is able to compensate against left and right displacements of image windows.

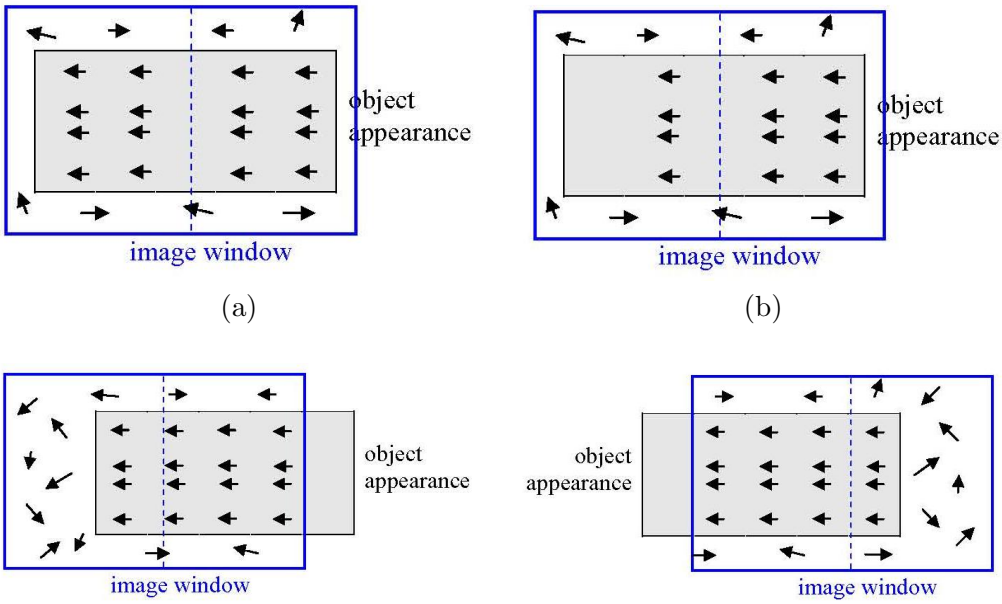


Figure 5.7: Illustrations of four possible cases at time $t-1$ when the Bayesian framework estimate $\widehat{x}_t$ is to the left of $\widehat{x}_{t-1}$. The arrows represent the motion vectors of feature points within the image window: (a) $\frac{\mu_L}{\mu_R} \approx 1$ for leftward translational motion; (b) Leftmost feature points on the object are not tracked by the KLT tracker because these feature points disappear in the next frame, for rotational motion to the left, yielding $\frac{\mu_L}{\mu_R} < 1$; (c) Another possibility for $\frac{\mu_L}{\mu_R} < 1$ is if the image window is erroneously displaced to the left; (d) $\frac{\mu_L}{\mu_R} > 1$ if the image window is erroneously displaced to the right (cases (c) and (d) may be due to the error in estimating $\overrightarrow{M}$ during tracking).

5.3 Experimental Results

This section presents the experimental results of tracking individuals in dense crowds by our tracker, possibly the first such tracker for this application. Results from two other trackers which are the closest fit to our problem, are also presented here for comparison: uniformly weighted feature-based tracker and mean shift tracker. Uniformly weighted feature-based tracker assumes rigid body motion and estimates motion simply as the average, $\overrightarrow{M} = \frac{\sum_i \overrightarrow{m_i}}{N}$. For the mean shift tracker, several region-based features can be used (e.g. color, texture or edges) to build the histograms; we chose color as the most suitable for tracking in cluttered backgrounds, just as in the original approach [21].

A total of 44 video sequences in 14 different scenes have been tested, with length ranging from 37 to 220 frames. Our tracker was able to track at 3 frames per second on video sequences of resolution 640 x 480, using a single core 3.4GHz PC with 3GB memory. As for our detector (refer to Chapter 4), the smallest detectable head resolution is 20 x 20 pixels. Hence, we set the number of feature points to be tracked within an image window of 20 x 20 to be $N = 40$. For them to be as uniformly distributed as possible within the window, we set the minimum distance between all feature points to be 3 pixels. The experimental values of q and τ are set to 10 and 2, respectively. In the first two subsections, we present results from the tracker which assumes only translational motion; in the following subsection, results of our modified tracker to handle scaling and rotation are presented in challenging scenes where objects exhibit scaling and rotational motions; then quantitative measures in evaluating both the trackers are described in the last subsection.

5.3.1 Tracking in Dense Crowds

Figures 5.8 to 5.13 show the familiar video sequences, where the initializations of the detection windows in Frame 0 (i.e. $t = 0$) have been obtained from our detector

described in Chapter 4. This subsection presents the results of the uniformly weighted feature-based tracker, and the mean shift tracker, and compare their performances to our Bayesian tracker:

Uniformly Weighted Feature-based Tracker

As shown from Figures 5.8 to 5.13, the uniformly weighted feature-based tracker is able to track in these video sequences, most of the time. But towards the end of the video sequences, some image windows have been displaced from the heads that were being tracked, such as the last frames in Figures 5.9, 5.11 and 5.12. This approach simply computes the average of all feature point motions even though some of the feature points may fall outside the head region, such as the chest or shoulder and background whose motion may be different from the head, resulting in the image window being displaced (as in Figure 5.9). This is especially so for the video sequences from Figures 5.11 to 5.13 where people surrounding the tracked individual may be moving in different directions. Hence, displacements of the image windows can be observed in two of these three video sequences (i.e. 5.11 and 5.12). We conclude that rigid motion tracking by averaging the feature motion vectors does not perform well here.

Mean Shift Tracker

From Figures 5.8 to 5.13, it can be seen that the mean shift tracker fails to track in most cases, except when an individual's clothing with respect to a neighbor is distinct, e.g. the person in pink and the person in green, respectively, in Figures 5.8 and 5.9. This implies that region-based approaches such as the mean shift tracker appear unsuitable for tracking individuals in dense crowds, since individuals are always very close to one another and have no simple, distinct region-based features for reliable tracking by such methods.

Bayesian Feature-based Tracker

The tracker we propose is generally able to track a given individual in the dense crowd throughout each of the six video sequences, as shown in Figures 5.8 to 5.13. Unlike the uniformly weighted tracker, it is able to track even with head and shoulder articulations, and noisy backgrounds within the image windows; unlike the mean shift tracker, it does not require the tracked object to have distinct spatial features from neighboring objects. Furthermore, Figures 5.8 to 5.10 specifically illustrate that initializations of our tracker can be made on the back views of heads and it is possible for it to track even though a feature-based approach is used. This is because we have set a high value of N to ensure that the distribution of the feature points within the image window is as uniform as possible, so that feature points are also detected on the back of the heads even though they are less textured (as long as the regions are not uniformly textured or colored as in cartoon images). Next, Figures 5.11 and 5.12 show that our tracker is able to track individuals who are stationary or move slowly on cluttered backgrounds even though neighboring individuals have similar appearance. This is because we constrain the tracker to a small search range for estimating $\widehat{x}_t$, in our Bayesian estimator and use linear approximation only when required. Finally, Figures 5.12 and 5.13 each shows an individual whose head rotates as he moves, but they are still tracked despite assuming only translational motion. We shall further look into this in Section 5.3.3 where results of individuals undergoing larger rotational motions are presented.

Our tracker was also used to track individuals in the data set of [90] with a resolution 480 x 360. Since our detector in Chapter 4 was not trained for a Caucasian crowd, image windows were manually initialized in Frame 0 as the inputs to the tracker, as shown in Figures 5.14 to 5.16. The scenario here is more challenging due to the lower resolution and with individuals walking in an unconstrained manner, but the tracker can still track the individuals reliably. Again, the reason is because we assume

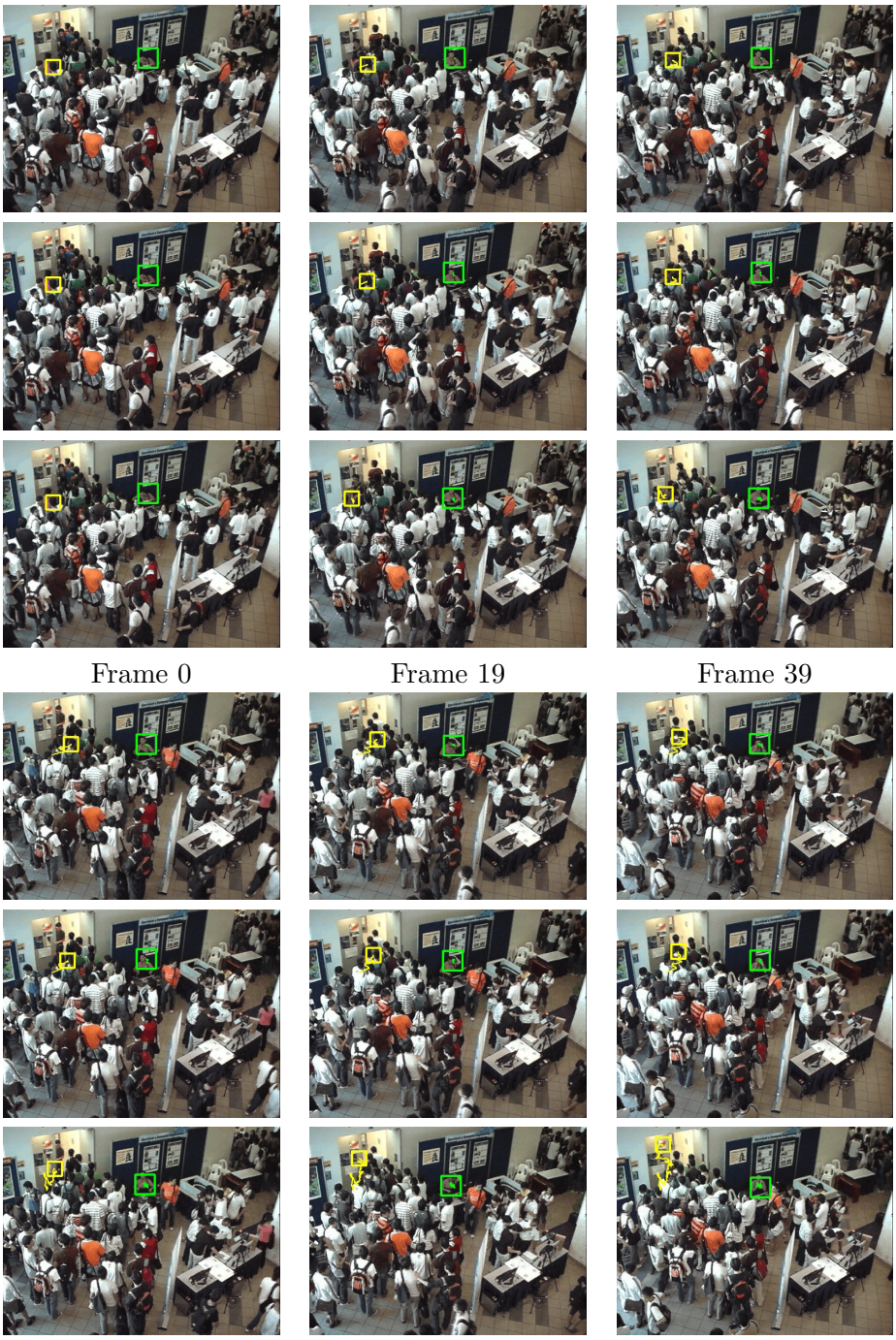


Figure 5.8: Tracking of first pair of individuals in a dense crowd exiting a common area through a doorway (upper is our proposed method, middle is uniformly weighted feature-based tracking approach, lower is mean shift tracking approach).

Figure 5.9: Tracking of second pair of individuals in a dense crowd exiting a common area through a doorway (upper is our proposed method, middle is uniformly weighted feature-based tracking approach, lower is mean shift tracking approach).

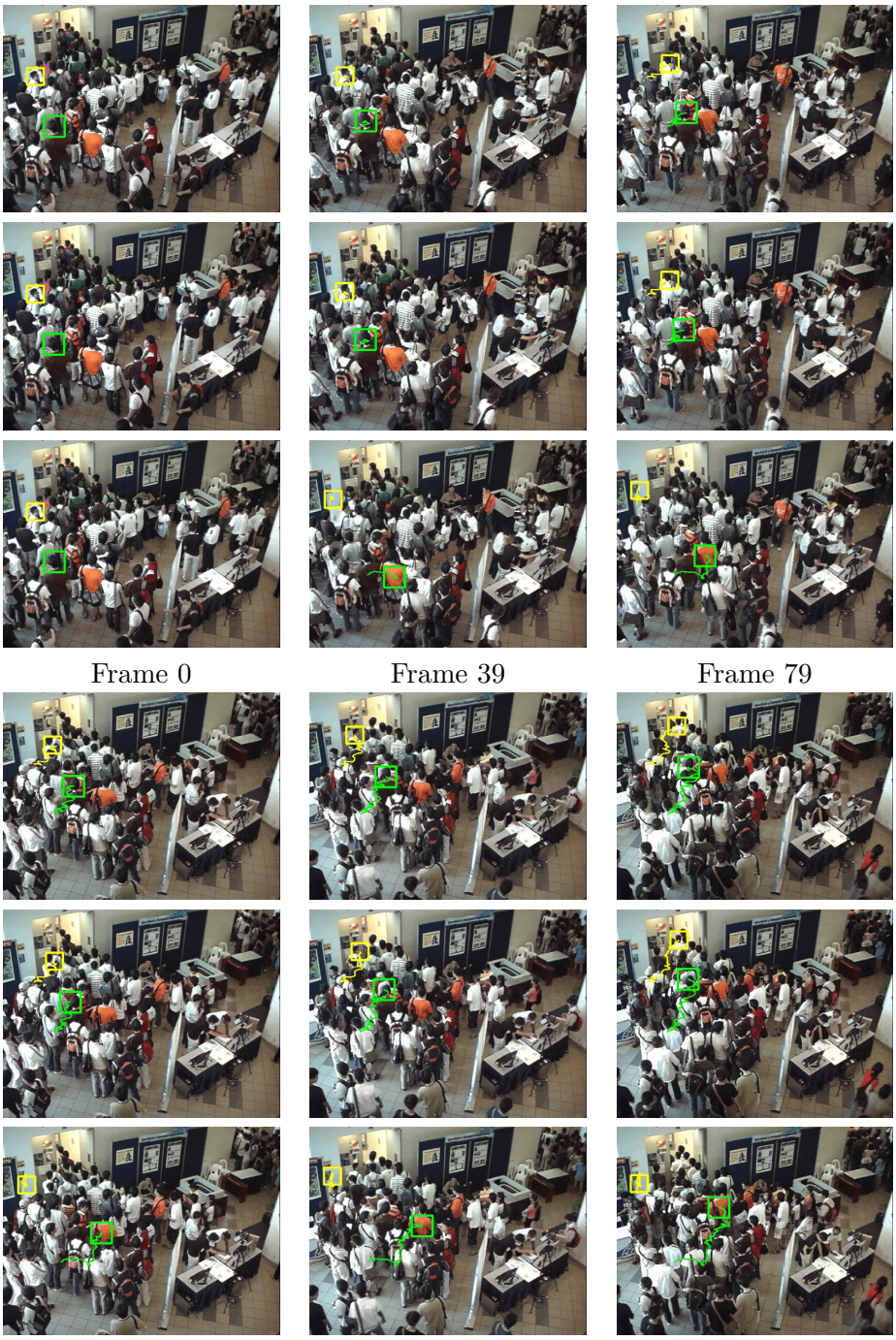


Figure 5.10: Tracking of third pair of individuals in a dense crowd exiting a common area through a doorway (upper is our proposed method, middle is uniformly weighted feature-based tracking approach, lower is mean shift tracking approach).

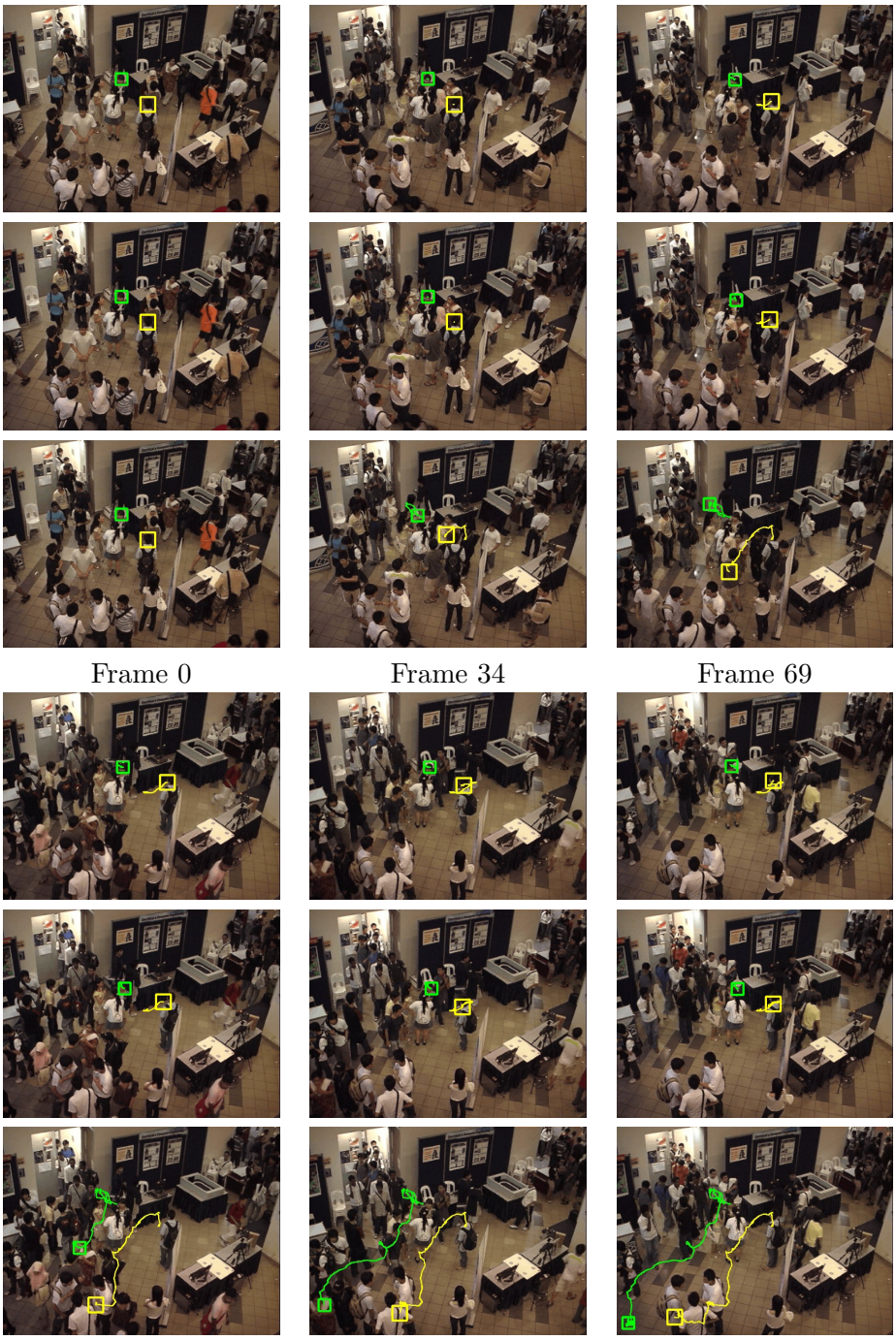


Figure 5.11: Tracking of first pair of individuals in a dense crowd exiting a common area through a doorway (upper is our proposed method, middle is uniformly weighted feature-based tracking approach, lower is mean shift tracking approach).

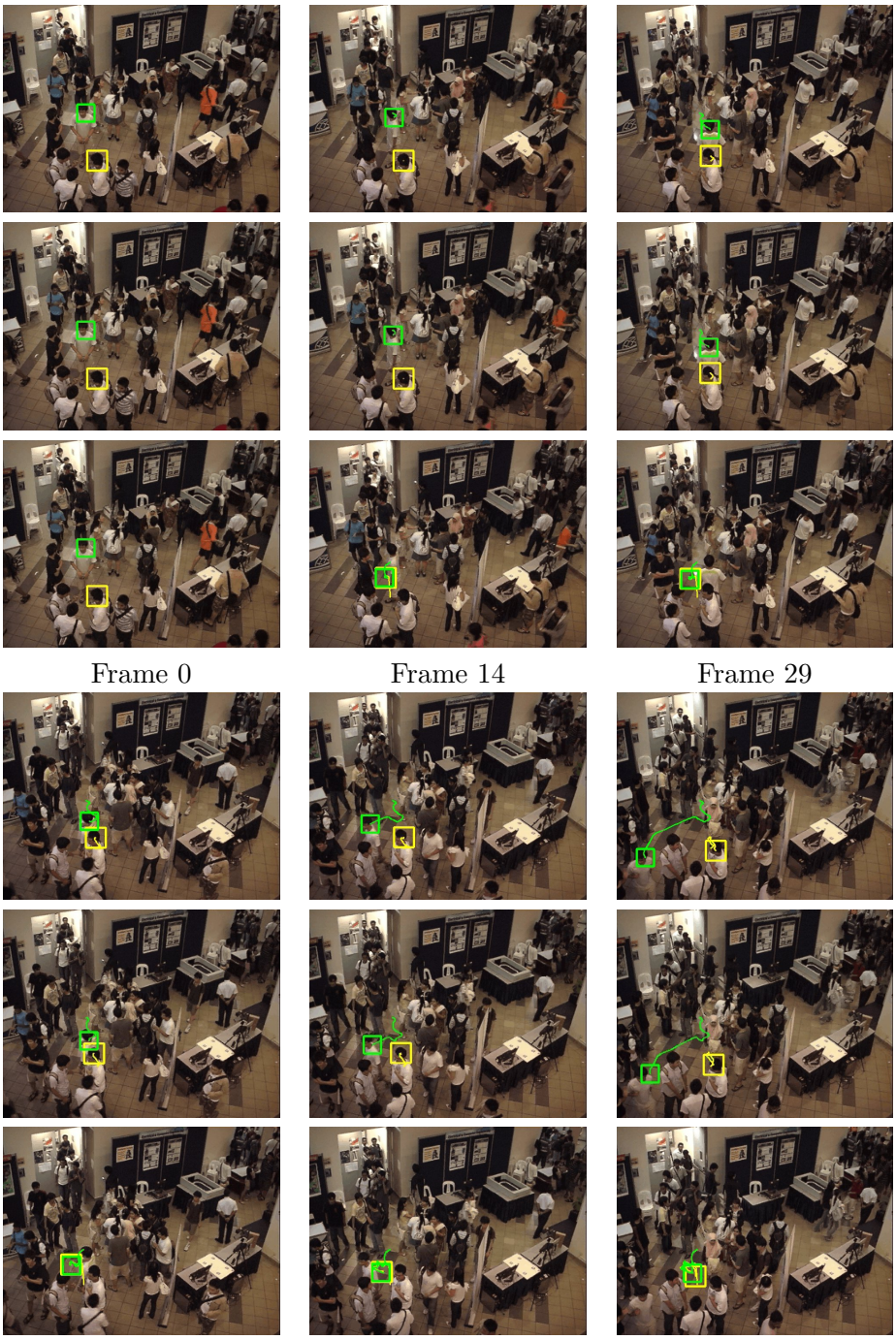


Figure 5.12: Tracking of second pair of individuals in a dense crowd exiting a common area through a doorway (upper is our proposed method, middle is uniformly weighted feature-based tracking approach, lower is mean shift tracking approach).

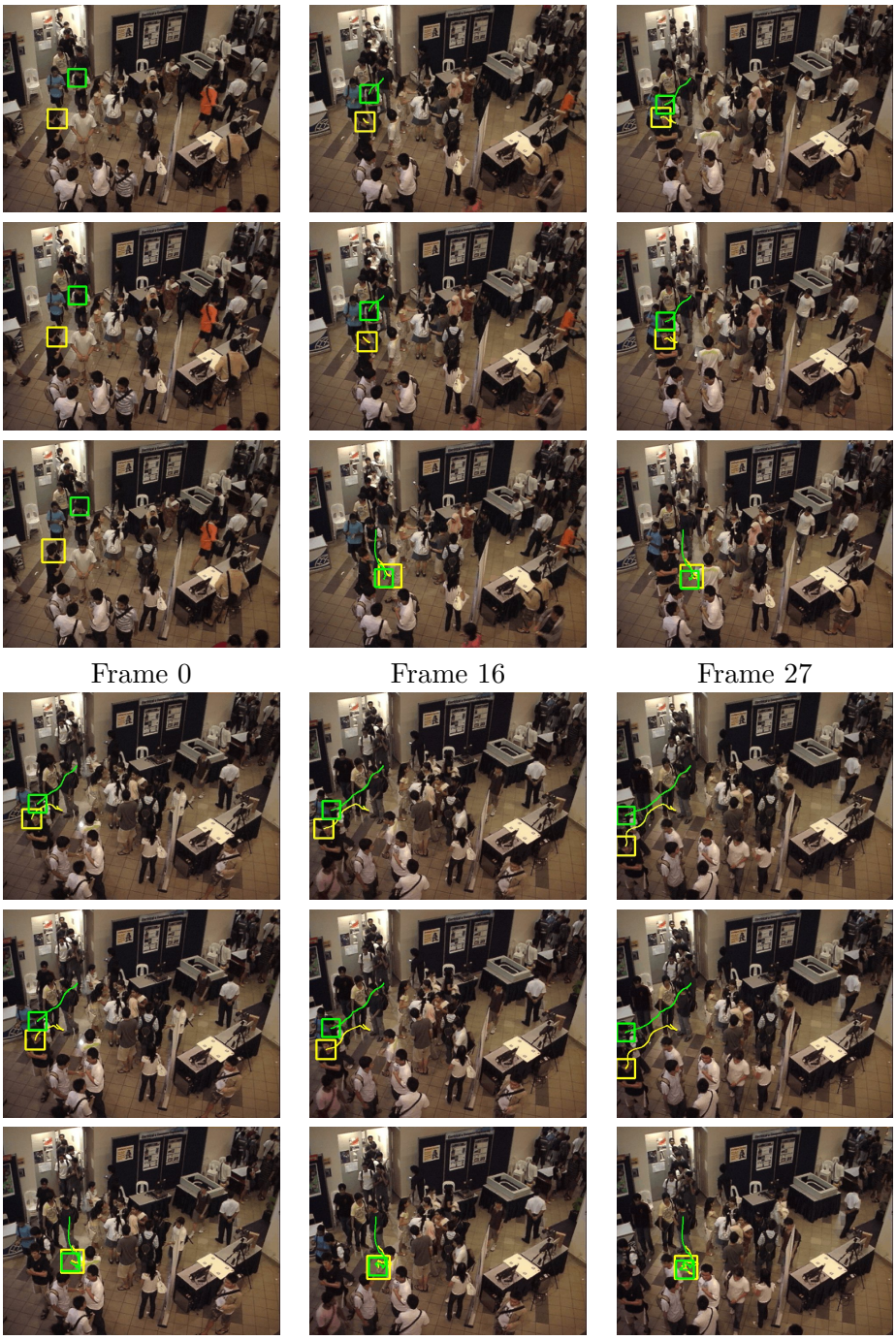


Figure 5.13: Tracking of third pair of individuals in a dense crowd exiting a common area through a doorway (upper is our proposed method, middle is uniformly weighted feature-based tracking approach, lower is mean shift tracking approach).

the individuals' motions between frames to be negligibly small and hence the search region to estimate $\widehat{x}_t$ is much constrained, so that the tracker is not easily confused by individuals of similar appearance. In particular, Frame 13 of Figure 5.16 shows a tracked individual being partially occluded but the tracker continues to track him reliably. This is due to the reciprocal weights used for the feature points and the motion vectors in the definitions of the prior probability and the likelihood, respectively (refer to Section 5.2.1).

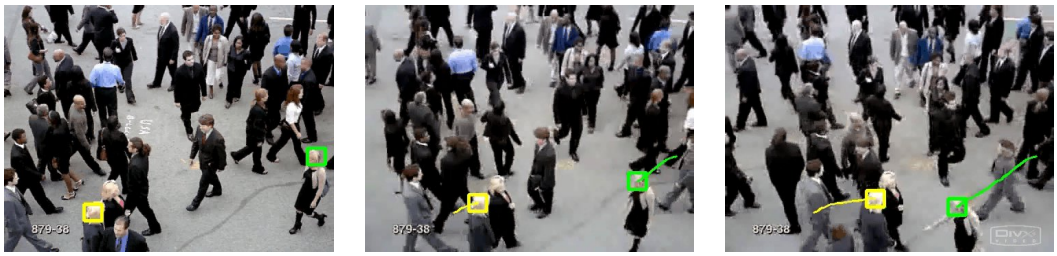
Frame 0 Frame 20 Frame 41

Frame 62 Frame 83 Frame 98

Figure 5.14: Tracking of first pair of individuals in dense crowds under unconstrained motion.

5.3.2 Tracking in Other Scenarios

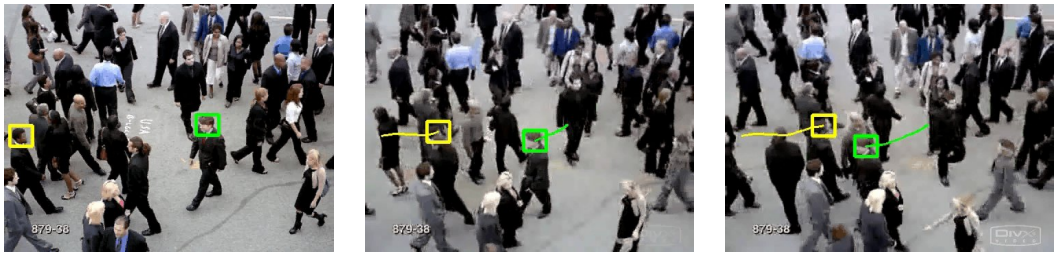
We tested our tracker on several other video sequences ranging from people to vehicles. The popular CAVIAR data set [50], is shown in Figure 5.17. The resolution of these video sequences is 384 x 288. We manually initialized windows to enclose the heads of the individuals which are as small as 20 x 20 for the first video sequence (the upper



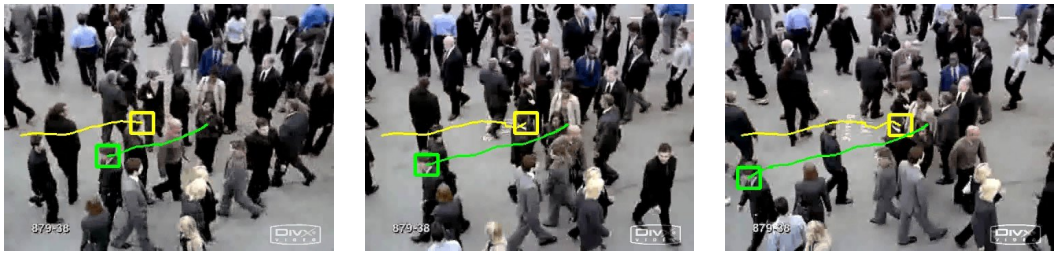
Frame 0 Frame 10 Frame 21

Frame 32 Frame 43 Frame 55

Figure 5.15: Tracking of second pair of individuals in dense crowds under unconstrained motion.



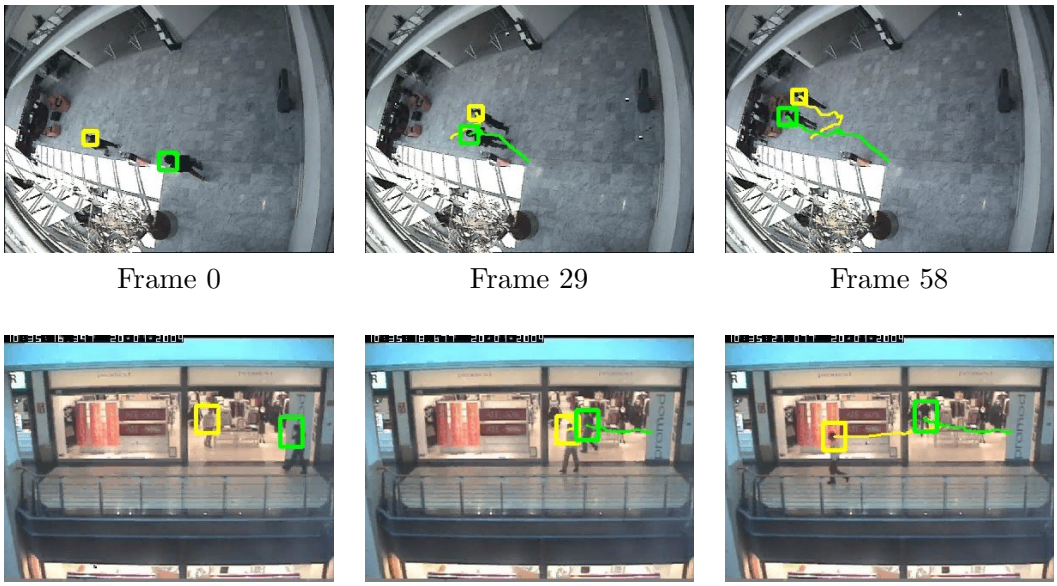
Frame 0 Frame 13 Frame 21



Frame 32 Frame 43 Frame 52

Figure 5.16: Tracking of third pair of individuals in dense crowds under unconstrained motion.

image sequence in Figure 5.17), to show our tracker's performance with respect to low resolution image windows. The second video sequence (the lower image sequence in Figure 5.17) demonstrates that the tracker keeps track of individuals walking in and out of a shop, though the background and lighting, change drastically.



Frame 0 Frame 29 Frame 58

Frame 0 Frame 24 Frame 48

Figure 5.17: Tracking of individuals in sample video sequences from CAVIAR data set [50].

Figures 5.18(a) and 5.18(b) show the results from a video where two individuals walk past each other. These video sequences have resolution 320 x 240. Regardless of whether the image windows are initialized on the individuals' heads or full bodies, our tracker performs equally well. This is because our approach is feature-based and not problem-specific, which also explains why our tracker works in Figure 5.18(b) even though the two individuals have the same colored clothing. More importantly, these two results demonstrate the performance of the linear approximator in our tracker, as there are instances of full occlusions in both cases.

Figures 5.18(c) and 5.18(d) show the results from video sequences with reso-

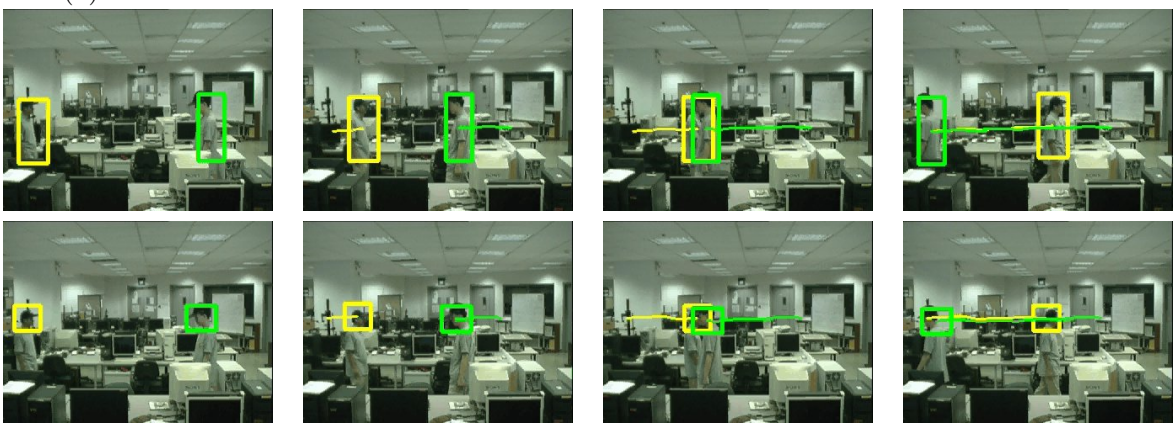
lution of 640 x 480. In addition to results presented in Section 5.3.1, Figures 5.18(c) and 5.18(d) show clearly, the performance of our tracker when only the back views of the individuals are seen and when only the front views of the individuals are seen, respectively. The sensitivity to scaling is also seen clearly here, in Frame 137 of Figure 5.18(d) where the image windows are too small for the individuals' heads because the individuals walk towards the camera, i.e. the chins of the individuals are not within the image windows as in Frame 0. In Section 5.3.3 we show results on individuals undergoing scaling, when we augment the tracker to handle this.

Finally, we present results on tracking vehicles in busy traffic scenes and also on soccer videos, in Figures 5.19 and 5.20 respectively. The video sequence of Figure 5.19(a) has resolution 720 x 404 and the video sequence of Figure 5.19(b) is 720 x 576; the video sequence of Figure 5.20(a) has resolution 720 x 576 and the video sequence of Figure 5.20(b) is 352 x 288. Figure 5.19 shows the different vehicles (motorcycles, cars and vans) that our tracker is able to track; Figure 5.20 shows our tracker's result on nonrigid motions (with much articulatory motions). As mentioned in Section 5.2.1, as long as the net translational motion is greater than the object's articulatory movements, our tracker is able to perform well. Hence, our tracker handles well both the rigid motion of vehicles as well as the nonrigid motion of the soccer players, in Figures 5.19 and 5.20 respectively.

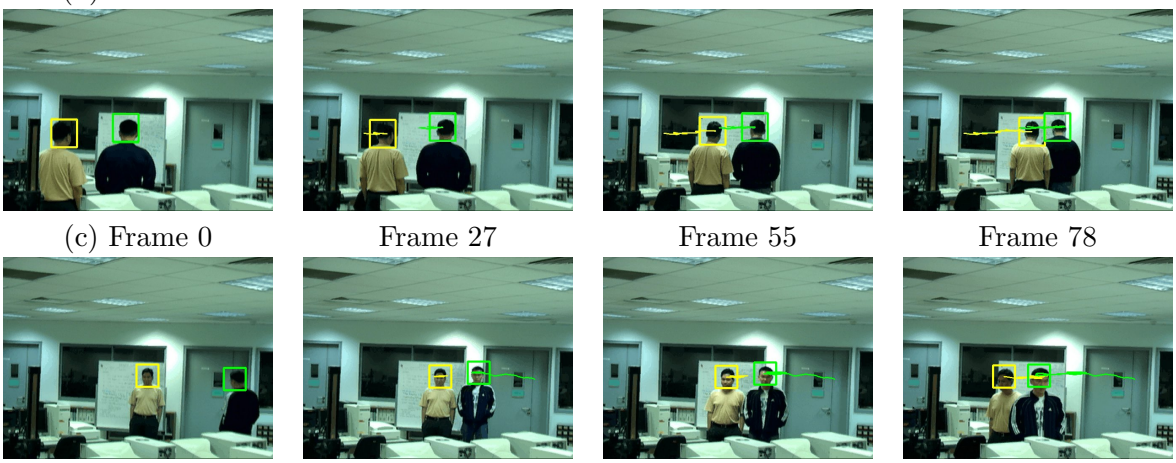
5.3.3 Tracking with respect to Scaling and Rotational Motions

This subsection presents the results when the tracker is modified to handle scaling and rotation as proposed in Section 5.2.3. In the context of this subsection, we denote $TrkV1$ as the tracker that only assumes translational motion and $TrkV2$ as the tracker which is designed to handle translational, scaling and rotational motions.

(a) Frame 0 Frame 14 Frame 29 Frame 43



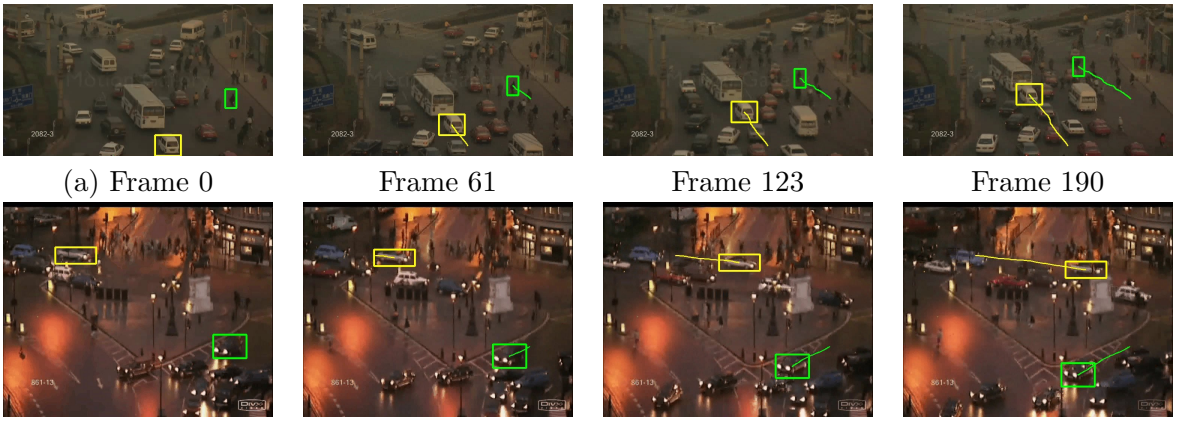
(b) Frame 0 Frame 11 Frame 23 Frame 40



(c) Frame 0 Frame 27 Frame 55 Frame 78

(d) Frame 0 Frame 49 Frame 99 Frame 137

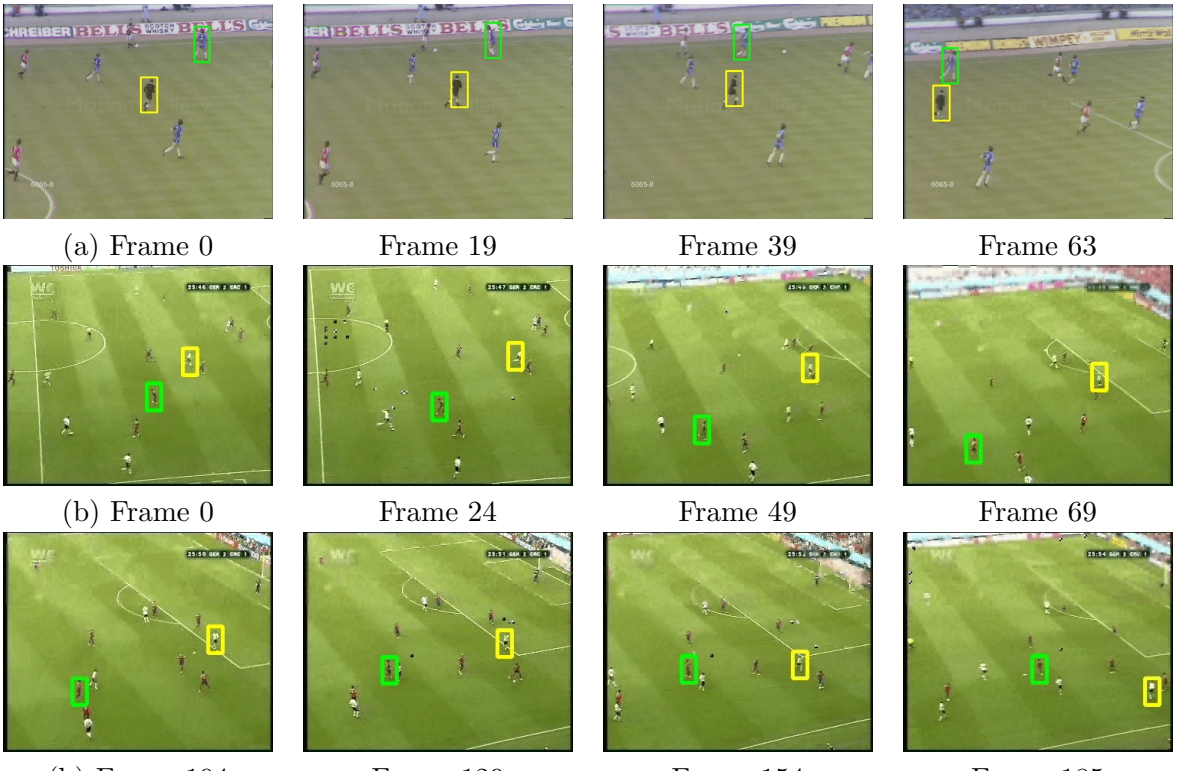
Figure 5.18: Tracking of individuals in simulated tracking scenarios: (a) Individuals are of different color clothing; (b) Individuals are of same color clothing; (c) Back views of individuals; (d) Front views of individuals.



(a) Frame 0 Frame 61 Frame 123 Frame 190

(b) Frame 0 Frame 29 Frame 59 Frame 83

Figure 5.19: Tracking of vehicles in sample video sequences from the data set in [90]



(a) Frame 0 Frame 19 Frame 39 Frame 63

(b) Frame 0 Frame 24 Frame 49 Frame 69

(b) Frame 104 Frame 129 Frame 154 Frame 185

Figure 5.20: Tracking of soccer players in sample video sequences: (a) From the data set in [91]; (b) Recorded from 2006 Federation International Football Association (FIFA) World Cup.

Robustness against Scaling Motions

Figures 5.8 to 5.16 show that when the individuals move closer or further away from the camera, the image windows become too small or too big for them, respectively. As mentioned previously, a more obvious scaling problem can be seen in Figures 5.18(c) and 5.18(d). With scaling taken into account, Figures 5.21 to 5.23 show the results of $TrkV2$. A clearer effect of the tolerance to scaling can be seen in Figure 5.24.

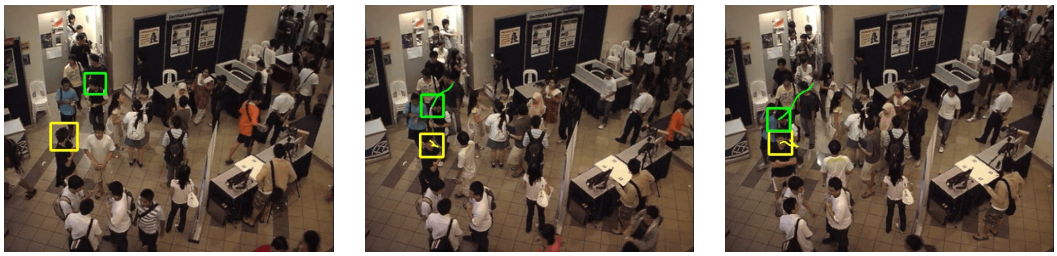
Frame 0 Frame 39 Frame 79

Frame 119 Frame 159 Frame 200

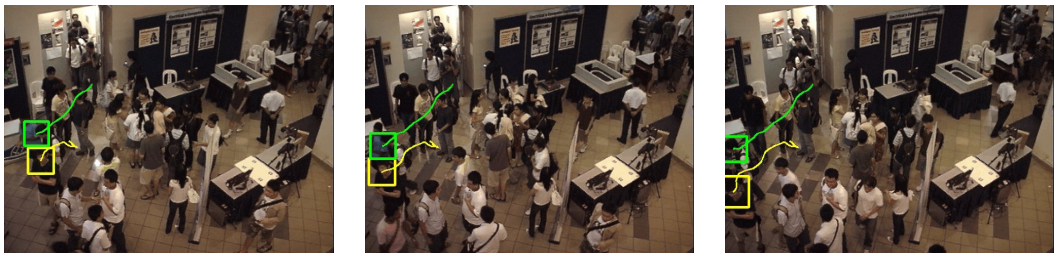
Figure 5.21: Tracking with respect to scaling on the video sequence in Figure 5.10.

Robustness against Rotational Motions

Here we closely examine the yellow image window in Figure 5.13, from Frame 0 to Frame 27 for $TrkV1$. Shown clearly in the upper row of images in Figure 5.25, the tracked individual's head is turning away from the camera initially, which results in the displacing of the image window, even though it was centered on the head at Frame 0. The reason why Figure 5.13 shows a reliable track of that individual using $TrkV1$ is



Frame 0 Frame 16 Frame 27



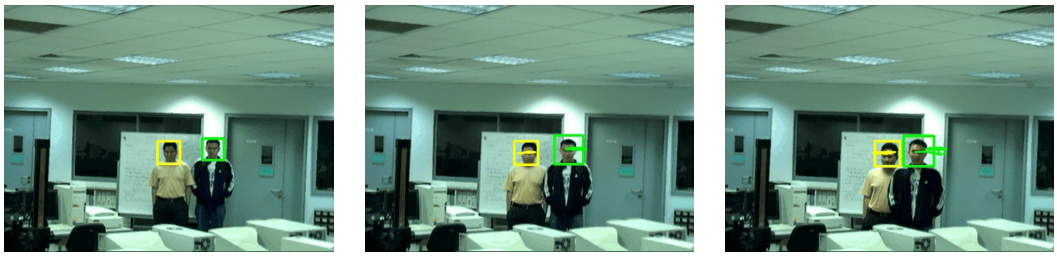
Frame 39 Frame 49 Frame 61

Figure 5.22: Tracking with respect to scaling on the video sequence in Figure 5.13.

Frame 0 Frame 20 Frame 41

Frame 62 Frame 83 Frame 98

Figure 5.23: Tracking with respect to scaling on the video sequence in Figure 5.14.



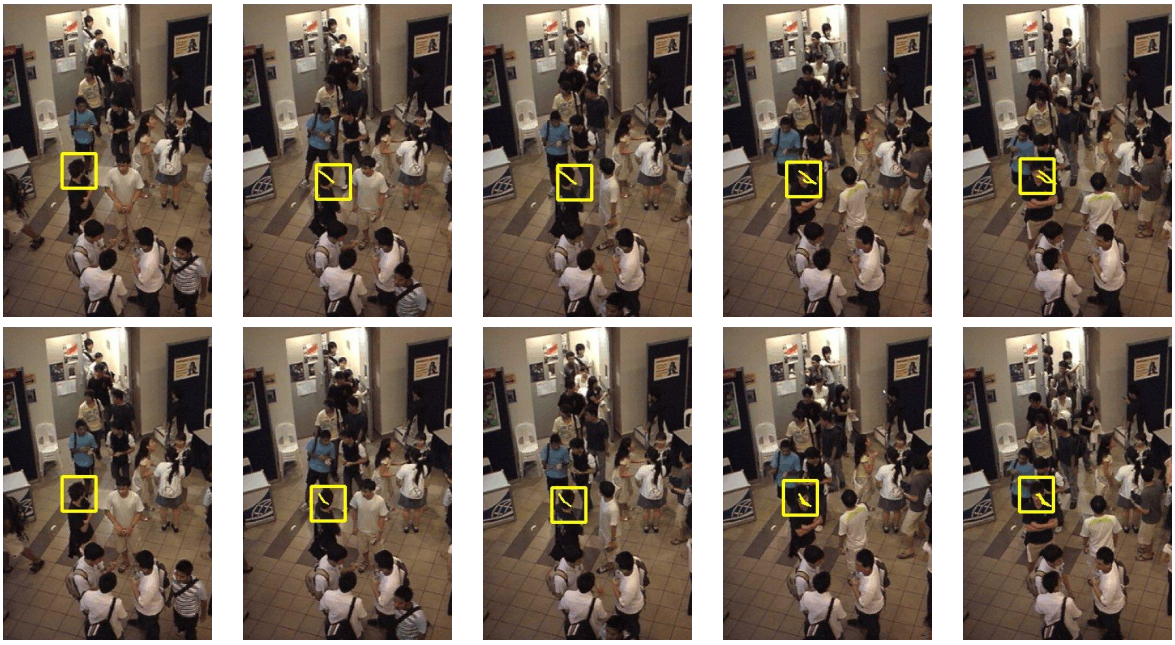
Frame 25 Frame 66 Frame 107

Figure 5.24: Tracking with respect to scaling on the video sequence in Figure 5.18(d).

because his head turns back after Frame 12, else mistracking may have easily resulted. On the other hand, the lower row of images in Figure 5.25, show that the image window is not displaced in any of the frames shown, demonstrating the improvement in tracking performance when handling objects under rotational motion is explicitly incorporated.

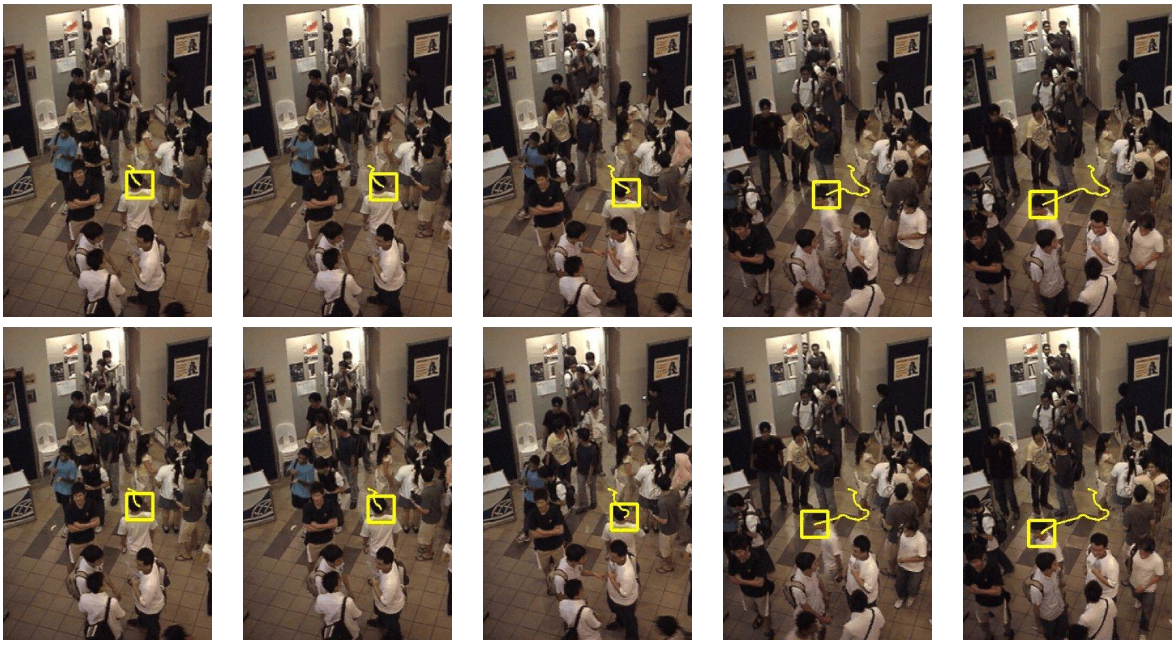
In Figure 5.26, the image window in Frame 22 is displaced to the right when the individual tracked is undergoing translational and rotational motion. As expected from the upper row of images, the image window of $TrkV1$ continues to be displaced to the right unless the individual's head rotates leftwards (but the individual's head rotates rightwards here). Hence, the image window is still displaced to the right of the head, though it remains tracked in Frame 60. On the other hand, the lower row of images shows that $TrkV2$ is able to handle such displacements by shifting the image window more to the left. But from Figure 5.26, the significant difference between the two rows of images is only seen from Frame 56 onwards. This is because the M_u values of the earlier frames are close to zero (see the tracking trails), where the effect of the additional characteristics is not as significant, until later frames when the individual translates leftwards.

A much clearer effect of the additional rotational characteristic can be seen in Figure 5.27. Using $TrkV1$, the image window displaces to the right because it infers that the rightwards rotational motion of the individual is a rightwards translation



Frame 0 Frame 6 Frame 12 Frame 18 Frame 24

Figure 5.25: Tracking of an individual under rotational motion in Figure 5.13 (upper shows the results of $TrkV1$, lower shows the results of $TrkV2$).



Frame 22 Frame 24 Frame 31 Frame 56 Frame 60

Figure 5.26: Tracking of an individual, whose image window is displaced in Frame 22, under rotational motion in Figure 5.12 (upper shows the results of $TrkV1$, lower shows the results of $TrkV2$).

motion. On the other hand, $TrkV2$ is able to handle this problem effectively.

5.3.4 Evaluation of $TrkV1$ and $TrkV2$

Here, we describe a quantitative evaluation procedure for the proposed trackers ($TrkV1$ and $TrkV2$), so as to systematically gauge the effectiveness of the trackers for all the tested video sequences. From Section 5.2, it is presented that the proposed tracking approach aims to estimate the CG of the current image window at every time instant, based on feature trajectories from the previous frame. Since an image window should encompass the head or the object to be tracked, the CG of the head or the object is a reasonable ground truth for the estimated CG of the image window, for every frame.

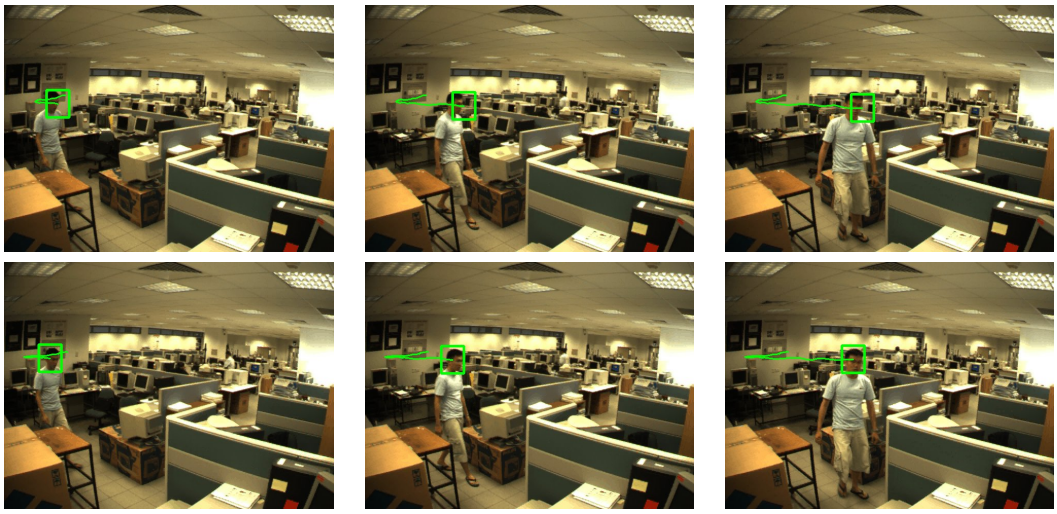
First, we manually mark the CG of the head or the object to be tracked, as the ground truth for all the evaluation frames. Then, the error, defined by the Euclidean distance in pixels between the estimated image window CG from the tracking approach and the ground truth, is recorded for each evaluation frame. Results are summarized in Table 5.1, where we evaluate every 10 frames in each video sequence of both $TrkV1$ and $TrkV2$.

	$TrkV1$	$TrkV2$
Sum of error in pixels	3731.06	1008.69
No. of evaluation frames	440	136
Error per frame in pixels	8.48	7.42

Table 5.1: Quantitative measure in evaluating $TrkV1$ and $TrkV2$.

From Table 5.1, it is observed that the error of both trackers is not more than 10 pixels for each evaluation frame, which is reasonably small because most test video sequences are of resolution 640 x 480. It is also observed that the error per frame of $TrkV2$ is more than 1 pixel lesser than that of $TrkV1$. Thus, it can be concluded here that both trackers perform well in the test video sequences, with $TrkV2$ being more

Frame 0 Frame 11 Frame 20



Frame 31 Frame 40 Frame 50

Figure 5.27: Tracking of an individual under rotational motion (upper shows the results of $TrkV1$, lower shows the results of $TrkV2$).

effective.

5.4 Summary

In this chapter we proposed a Bayesian approach to track individuals in dense crowds. The proposed tracker uses several KLT feature points and computes their weights in tracking the object via a Bayesian framework. The tracker also includes a linear approximator to handle significant occlusion problems. In addition, simple techniques to deal with scaling and rotation are proposed to improve the tracker's performance under such situations.

Results of the proposed tracking system have been presented on densely crowded scenes and compared with uniformly weighted feature based tracking and mean shift tracking, a simple tracker and a state-of-the-art tracker, respectively, that did not perform effectively in our crowded scenes. Also, several people tracking and vehicle tracking scenarios were presented using our approach. The proposed tracker not only tracks individuals in dense crowds well, it also performs equally well in many other tracking scenarios. The biggest drawback of our tracker is that it is feature-based with no recognition capability, and tracks whatever is initialized. Similar to mean shift tracking, if the image window is displaced from the object during initialization or when tracking, mistracking may result easily. That is why we proposed a linear approximator and incorporated additional characteristics into the tracker, which have shown to be effective in preventing displacement of the image window during difficult tracking situations (i.e. occlusions, scalings and rotations).

Chapter 6

Finding the Best Frontal Facial View

In this chapter, we propose to find the best frontal facial view of a person from among the multiple images in a video sequence, given the detection and tracking of his head as described in the previous chapters. Specifically, we aim to obtain the best possible view available in a video sequence to facilitate further processing, such as face recognition.

The application proposed here can be used for both single person and dense crowd video sequences. In each of these video sequences, given detection and tracking results of the person, the task of finding the frame with the best frontal pose is very much the same for both types of sequences. Here, we perform a thorough analysis on single person videos, apply the proposed approach to individuals in dense crowd videos.

This chapter first reviews the available approaches that are able to solve this problem, before describing our method. Next, experiments are presented which show promising results with respect to person dependency, low pixel resolution, occlusion problems and the ability to perform well in densely crowded scenes.

6.1 Works on Face Pose Estimation

To our knowledge, there is no work in the literature that deals explicitly with the problem considered here. However, estimation of face pose, which also plays an important role in HCI can be used to find the most frontal face, but it is computationally complex and hence we do not consider it. In the rest of this section, we briefly present previous works on face pose estimation, which can be appearance-based or model-based.

Appearance-based methods

Appearance-based methods use facial images as 2D patterns of image intensity, colors, gradients, etc. Often, a large number of training images are used to determine a mapping between the actual face pose in 3D and certain properties of the 2D facial images obtained. In [92], Okada and von der Malsburg use a parametric piecewise linear subspace method for differentiating face poses. Sherrah and Gong [93] investigate the correlations between head pose and the face-head displacement, by using a face detector and a head detector separately in 2D images. Wang and Sung in [94] and Liang et al. in [95] assume the projections of eye corners and mouth corners as vanishing points, from which the back projection calculations of these points are used to estimate the face pose. Li and Zhang [96] estimate face pose through FloatBoost Learning. Srinivasan and Boyer [97] use the eigenspaces of 2D images with different facial poses.

Model-based methods

Model-based methods usually assume a 3D head model to recover the face pose from 2D facial images, using 2D to 3D feature correspondences. These methods include use of bunch graphs for Elastic Graph Matching [98] and 3D appearance models [99], or simple geometrical models such as elliptical shape for the head with the positions of pupils known and assuming that gaze is frontal [100], cylindrical shape for the face [101]

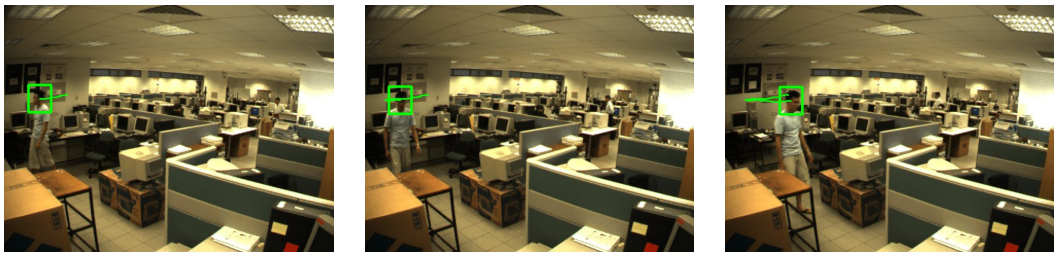
and an inverted triangle marking the eyes and mouth [102]. Other than using image intensity to construct the models, Malassiotis and Strintzis [103] have shown the possibility to construct a 3D head model for pose estimation using range data.

Though both appearance-based and model-based methods seem to show reasonably good and robust results even in low resolution facial images (i.e. 25 x 25 pixel resolution), they also have weaknesses. The appearance-based methods mentioned in [92] - [97], with the exception of [94] and [95], which depend highly on the accurate detection of facial features, all require large amounts of training data. Also, these methods including [94] and [95], are susceptible to motion blur, lighting variations, targets wearing glasses, targets with facial hair, targets with distinct facial expressions or target's face under occlusion. Similarly, model-based methods are susceptible to problems such as different hair styles and also to some of the problems listed for appearance-based methods. For example, the models proposed in [98] - [103] are built and experimented based on only a few persons, and are therefore not person independent.

6.2 Finding the Best Frontal Facial View

As mentioned, we aim to find the best frontal face pose from among the 2D images of a person's head, detected and tracked in a video sequence, using a simple technique that bypasses the complex face pose estimation. Assuming the target's head is provided in an image window for every frame as in Figure 6.1, our approach combines frame differencing with the skin color detector described in [104] to locate the skin region of the target's head. Next, we appropriately define two different rectangular bounding boxes, which encloses this skin region. For each image view of the target's head in the video sequence, the areas of first bounding box and skin region, and the CGs of second bounding box and skin region are calculated. These results are then used to calculate

a parameter, *Frontalness Value* that is an indicator of the best frontal view among all the image views. The rest of this section describes the procedure for computing the *Frontalness Value* in every image frame of the video sequence.



Frame 13 Frame 25 Frame 35

Figure 6.1: Image sequence showing target's head detected and tracked in an image window.

6.2.1 Calculating *Frontalness Value*

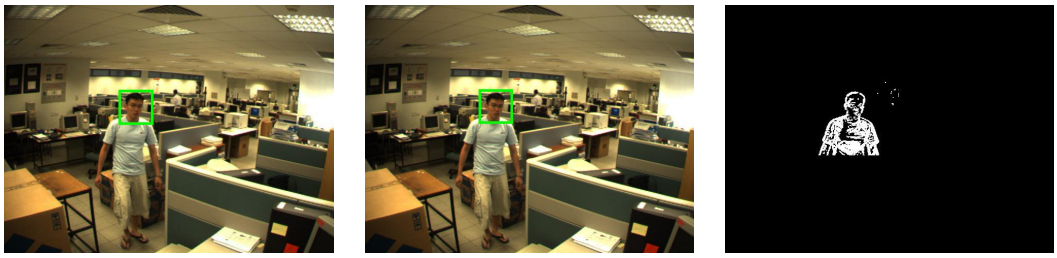
Frontalness Value aims to provide a measure for how 'frontal' the corresponding facial view is in relation to other facial views of the same person in a video sequence, the frame that yields the largest *Frontalness Value* is selected as the one with the best frontal face in the video. The novelty is in the use of a normalization strategy to make the measure independent of individuals, illumination changes, size of facial images and occlusion. We present the algorithm for computation of *Frontalness Value* in the following steps assuming that the target is not bald.

Step 1: Skin Region Detection

Our approach relies critically on two bounding boxes we define; each encloses the skin region of the target's head together with a portion of his hair in a different way. The motivation for defining these bounding boxes comes from the knowledge that the skin region of a head contains the most information needed for further face processing. This

approach has the potential to be extended to obtaining the best possible view of a person's face even under occlusion or different hair styles. The following describes the procedure for obtaining the skin region of the target's head.

Frame Differencing: Given the video sequence with the detected and tracked head, we first perform frame differencing on consecutive frames and then consider only the motion regions within the window enclosing the tracked head, and the nearby surrounding regions, as in Figure 6.2. Motion regions surrounding the target window are included, to account for the possibilities that the window may be too small for the head or displaced from the center of the head during tracking.



(a) (b) (c)

Figure 6.2: Frame differencing: (a) Video frame no. 278 of a fixed-lens camera; (b) Video frame no. 279 of the same camera; (c) Thresholded frame difference between (a) and (b) only in the regions near the target window.

Morphological Operations: Simple erosion and dilation operations are applied to remove noise and enhance the motion region of interest as in Figure 6.3(a), to obtain a motion silhouette.

Skin Color Detection: We perform skin color detection within the motion silhouette, based on the skin color detector in [104]. Here the RGB color components are transformed to *Luminance-Chrominance* (YC_rC_b) color space, as skin colors of different races are found to occupy a compact region of C_rC_b color space. We used the skin color region defined in [104] for our work. See Figure 6.3.

The skin region closest to the target window is taken to be the facial skin region.

(a) (b) (c)

Figure 6.3: Skin color detection: (a) Motion silhouette obtained after simple morphological operations on Figure 6.2(c); (b) Skin color detection in the motion region in (a); (c) Result of (b) after morphological operations.

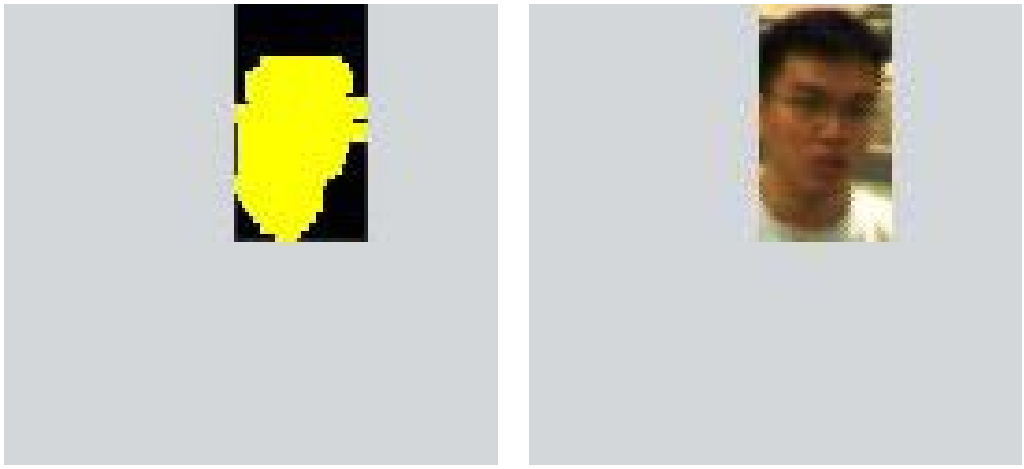
Step 2: Skin Area Ratio

Next, we define a rectangular bounding box for the skin area obtained from Step 1 as in Figure 6.3(c). This box, which encloses the skin region of the target's head together with a portion of his hair, is the smallest rectangle that encloses the face region, but its upper edge corresponds to the top of the motion silhouette as in Figure 6.3(a). Figure 6.4 shows pictorially the procedure for extracting this bounding box in each image frame.

Next, a *Skin Area* ratio, is defined as $R_1 = SArea/RArea$ where $SArea$ is the area of skin region in the bounding box, $RArea$ is the area of the bounding box, to measure the proportion of facial skin within the box. This ratio measure is used to induce invariance to the size of faces in images. The higher this ratio is, the likelier it is that the camera is viewing a frontal face pose. See Figures 6.5(a) to 6.5(d).

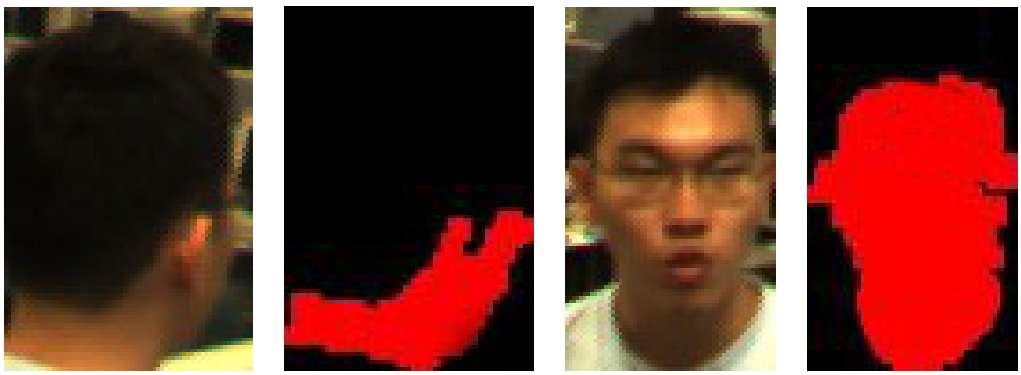
Step 3: Horizontal Displacement from the Center

Similar to [105], we define a second bounding box that encloses the head by making use of the motion silhouette together with the chosen skin region. In comparison with



(a) (b)

Figure 6.4: Bounding box 1: (a) The bounding box obtained based on the skin regions in Figure 6.3(c) near the tracked window; (b) Corresponding face within the bounding box.



(a) (b) (c) (d)

Figure 6.5: Skin area ratio: (a) Face is turned away from the camera; (b) Only a small portion of the bounding box is detected as skin region; (c) Frontal face view in a high resolution image under normal lighting; (d) A large portion of the bounding box is detected as skin region.

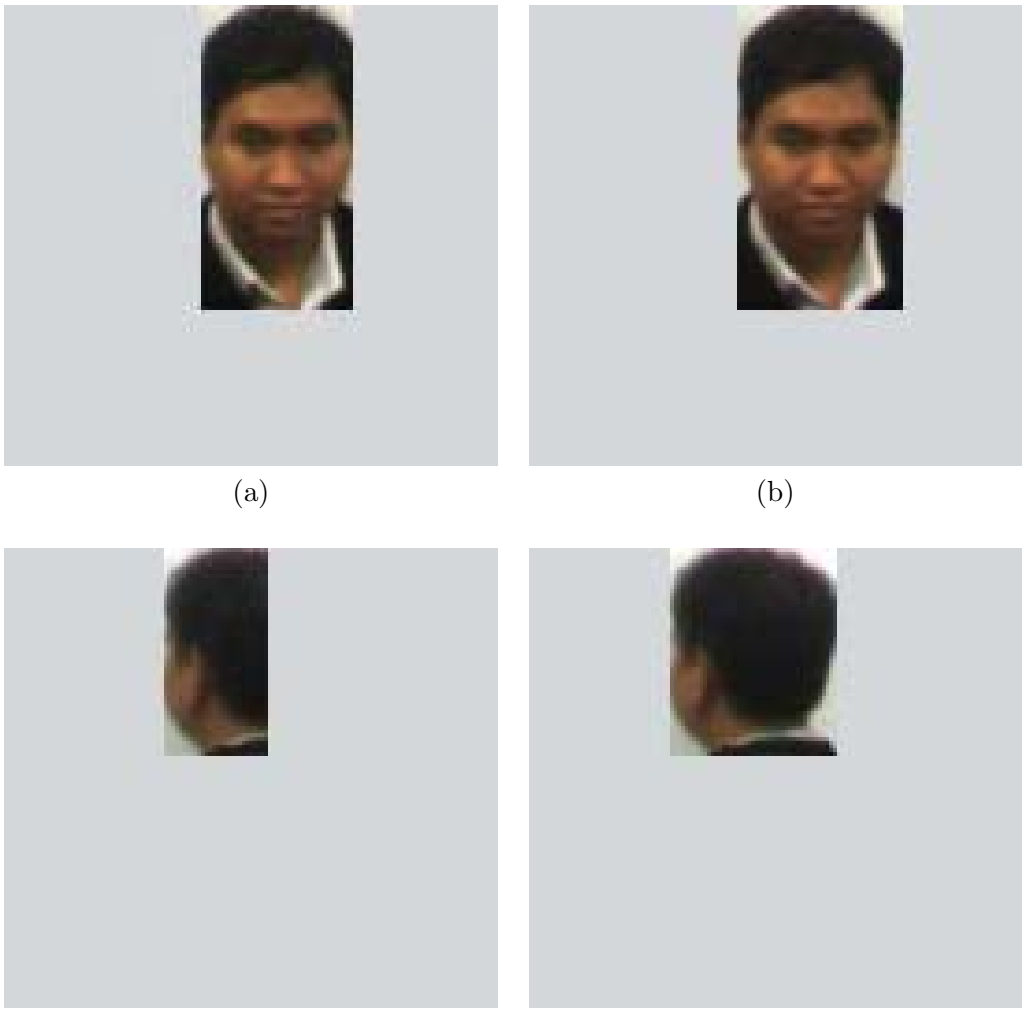
the first bounding box, the second one is of the same height but of different width as it encloses all of the hair and skin regions of the target's head. See Figures 6.6(a) to 6.6(d) for a pictorial comparison between the two bounding boxes.

Then we find the CGs of the skin region in this bounding box and of the bounding box itself, and denote their x-coordinates $xSkinCG$ and $xRectCG$, respectively. We define a parameter $R_2 = |xRectCG - xSkinCG|/W$. The normalization by W, the width of the bounding box makes R_2 insensitive to face size. R_2 indicates how much the CG of the skin region is displaced horizontally from the center of the bounding box, and is therefore able to estimate how close to frontal the face pose is, assuming that the skin region of a face is symmetric when frontal [105]. See Figures 6.7(a) to 6.7(d).

Step 4: Frontalness Value

The *Frontalness Value*, F, is used to specify the best frontal view of a target's face from among the frames in a video sequence. For each image frame, this measure is calculated by combining the two parameters, R_1 and R_2, defined above. As mentioned, larger R_1 values or smaller R_2 values will favor more frontal faces, and F is therefore defined simply as $F = R_1 - R_2$ (see Appendix A for the verification of this definition). Once F is calculated for each image frame, the frame that yields the largest value of F is chosen as the best frontal view of the person in the video sequence. The algorithm below summarizes Steps 2 to 4.

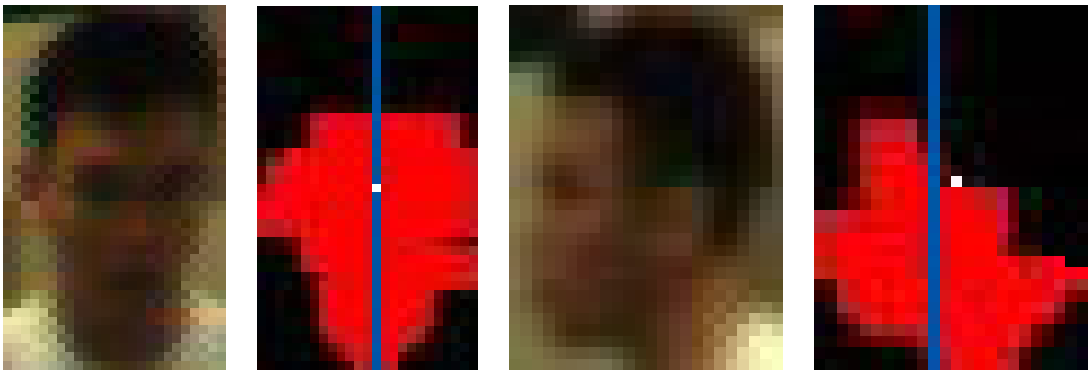
The reliability and robustness of our approach, which is independent of facial features, is much better compared to methods that depend highly on facial features, because they can be difficult to locate. Low resolution images or features blurred by motion is no more a concern because area ratios and CG are used here. Also, our approach is not sensitive to people wearing glasses, facial expression changes, normal lighting vari-



(a) (b)

(c) (d)

Figure 6.6: Bounding box 1 and 2 comparison: (a) First bounding box for example 1; (b) Second bounding box for example 1; (c) First bounding box for example 2; (d) Second bounding box for example 2.



(a) (b) (c) (d)

Figure 6.7: Horizontal displacement from the center: (a) Frontal face view in a low resolution image under weak lighting; (b) $xSkinCG$, which is shown by the vertical line, passes through the CG of the bounding box, which is represented by the white dot; (c) Profile view of the face; (d) $xSkinCG$ now displaces horizontally from the CG of the bounding box.

ations in a room or different hair styles as it is not model-based. Most importantly, our approach is person independent and requires no training. It has the potential to resolve other common problems faced by appearance-based and model-based approaches, such as targets with facial hair and occlusion. Experimental results are shown in the next section to justify the above statements.

6.3 Experimental Results

Here, each video sequence is 500 frames in length and shows a person walking in an unconstrained manner within the camera FOV. The video is captured at a frame rate of 30 frame per second, with each frame digitized to 640 x 480 pixels with 8-bit precision for each color channel. Our experimental computer runs on a Pentium IV 2.4 GHz processor; it is able to process real-time in finding the best frontal facial view of the person as he walks, although our experiment here is an offline process to investigate accuracy.

Algorithm 3 Selecting best frontal facial view in a video sequence.

for all $v \in V$ **do** ▷ V refers to the set of all image frames.

$R_1(v) := SArea(v)/RArea(v)$
▷ ratio of skin area to bounding box 1 area.

$R_2(v) := |xRectCG(v) - xSkinCG(v)|/W(v)$
▷ normalized horizontal-displacement from CGs.

$F(v) := R_1(v) - R_2(v)$

end for

$v_f := arg_v \max F(v)$ ▷ v_f corresponds to image frame with best frontal face.

6.3.1 Basic Setup Scenario

For experimental analysis, we randomly select two pairs of consecutive frames from different time instants in the video sequence. For each pair of consecutive frames, the methodology described in Section 6.2.1 is applied to obtain the *Frontalness Values*, F. Then, the pair of consecutive frames with the higher F is selected as the best frontal facial view for this selection. See Figure 6.8 for an example.

We repeated the above test 500 times, and found that the algorithm yielded the pair with the more frontal face in 470 instances for an accuracy of 94.0%. We used human judgement to obtain the ground truth for 'best frontal view' (e.g. see Figure 6.8).

Of the 30 wrongly chosen images, 24 of them resulted from failure to obtain a frame difference signal which occurs when the target is stationary or moves extremely slowly at the time of frame capture. In the remaining instances, though good frame differencing signal is available, the resolution of the target's face in the image is very low, causing the error. This high accuracy rate shows that our approach is robust with respect to complex background environments and normal lighting variations in

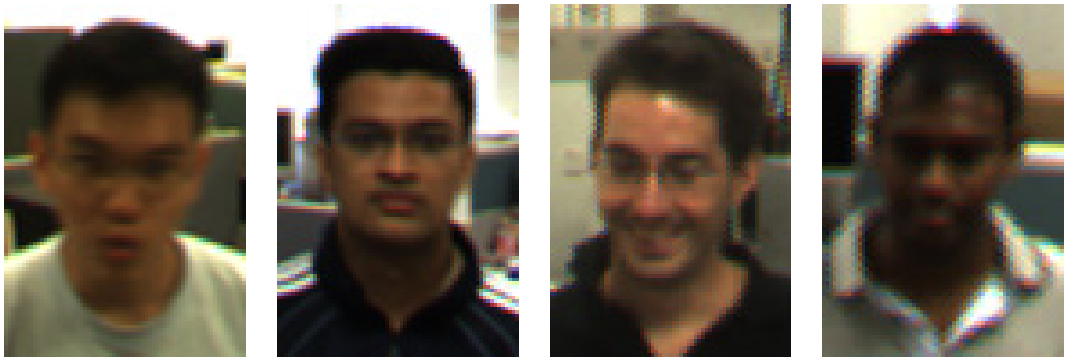
Figure 6.8: Pairs of consecutive frames from a video sequence (upper shows Frames 168 and 169 with $F = 0.322$, lower shows Frames 278 and 279 with $F = 0.551$).

the room, as evident in Figure 6.8.

6.3.2 Robust Features of Our Approach

Person Independence

To check that our approach is person independent without using any training, we experimented with persons of different skin color, since our method is based on skin color detection. Figure 6.9 shows the four persons we selected to test for person independence.



(a) (b) (c) (d)

Figure 6.9: Four persons used to test for person independence: (a) Person A (Chinese); (b) Person B (South Asian); (c) Person C (European); (d) Person D (South Asian).

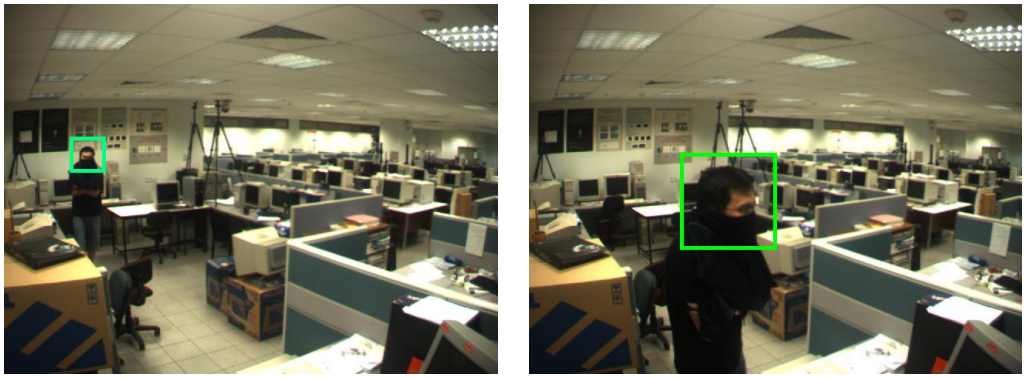
As described in Section 6.3.1, each of these four persons walked in an unconstrained manner within the FOV of the camera individually. The results of our algorithm are shown in Table 6.1, which shows that the approach is independent to skin color differences and person identity. The latter property follows because the comparison of computed F values is done for each individual, and does not depend on the F values of others.

Person	A	B	C	D
Accuracy Rate %	98.8	98.8	98.8	99.2

Table 6.1: Accuracy rate for 4 different persons from video sequences where each is 500 frames in length.

Robustness against Occlusion

Here, we investigate whether our approach has the potential to deal with occlusion problems, such as targets with facial hair or targets with different hair styles. Person A in Figure 6.9 has covered the lower half of his face with a towel, and the experiment as in Section 6.3.1 is repeated, as depicted in Figure 6.10, in a video sequence that is 500 frames in length. This experiment yielded perfect result with 100% accuracy. Hence, we believe that our approach has the potential to deal with more complicated occlusion problems of the face.

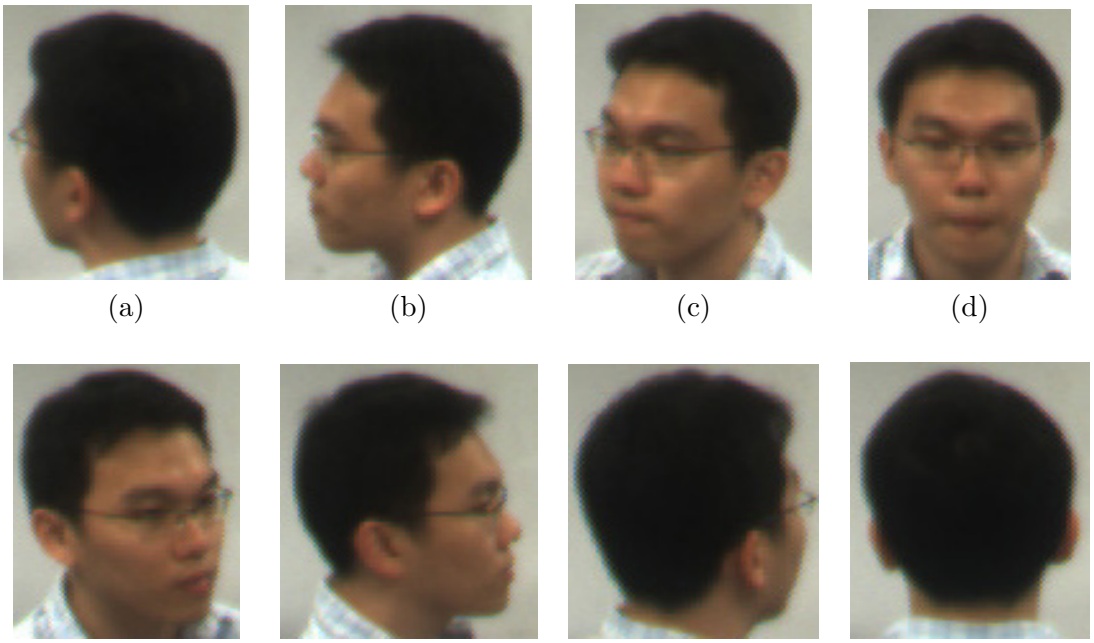


(a) (b)

Figure 6.10: Occluded face experiment: (a) Frame 224 showing frontal view of target; (b) Frame 348 showing profile view of target.

Tolerance to Size of Facial Image

Previously in Figures 6.5 and 6.7, it could be pictorially observed that skin detection of facial images works well with different image resolutions and lighting conditions. This implies that the value of F should not be affected much, when the camera captures the same facial view of the target from different distances. Here, we examine the tolerance of our approach to different face sizes in images, using a total of 32 facial view images consisting of eight different facial poses (i.e. $0°$, $\pm 45°$, $\pm 90°$, $\pm 135°$, $+180°$) of a person, for each of four face sizes from 80 x 100 to 12 x 16. Figures 6.11(a) to 6.11(h) show the views for face size of 80 x 100. Then, F is computed for each image, where only steps 2 to 4 of Section 6.2.1 is applied. Table 6.2 shows the results.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 6.11: Facial Poses of 80 x 100 pixels resolution: (a) $-135°$; (b) $-90°$; (c) $-45°$; (d) $0°$; (e) $+45°$; (f) $+90°$; (g) $+135°$; (h) $+180°$.

All the four sizes have their peak F values at $0°$ facial pose and decrease as the

	Pixels resolution			
Facial pose	80 x 100	40 x 60	20 x 24	12 x 16
$-135°$	-0.075	-0.100	-0.087	-0.072
$-90°$	0.216	0.225	0.156	0.101
$-45°$	0.357	0.352	0.309	0.297
$0°$	0.429	0.412	0.433	0.394
$+45°$	0.379	0.375	0.334	0.282
$+90°$	0.200	0.225	0.120	0.133
$+135°$	-0.054	-0.036	0.027	-0.059
$+180°$	0.017	-0.017	-0.055	-0.205

Table 6.2: F values of different facial poses at four different sizes from 80 x 100 to 12 x 16 pixels resolution.

facial pose deviates from $0°$, except for the $+180°$ facial pose which is slightly unstable. Since the $+180°$ facial pose has no facial information and its F value is relatively small, its variations will not affect finding the best frontal facial view. From the results in Table 6.2, we can conclude that our approach is indeed size invariant to faces as small as 20 x 24 pixels since the values of F from images smaller than 20 x 24 pixels are relatively less well-behaved for most facial poses. The reason may be due to color uncertainty of facial edge pixels which become an influencing factor as images become small.

6.3.3 Dense Crowd Scenario

This subsection presents the results of the algorithm on the tracked video data from Chapter 5, particularly on the dense crowd videos in Figures 5.8 to 5.13. First, the yellow or green image windows (tracking results) of each tracked individual are cropped out from the frames of the video sequence. Then, we applied steps 2 to 4 of Section 6.2.1 on the cropped images and chose the one with the highest F as the best frontal facial view of that individual. Figure 6.12 presents the results.

Among the individuals in the dense crowd videos, the three individuals in Fig-

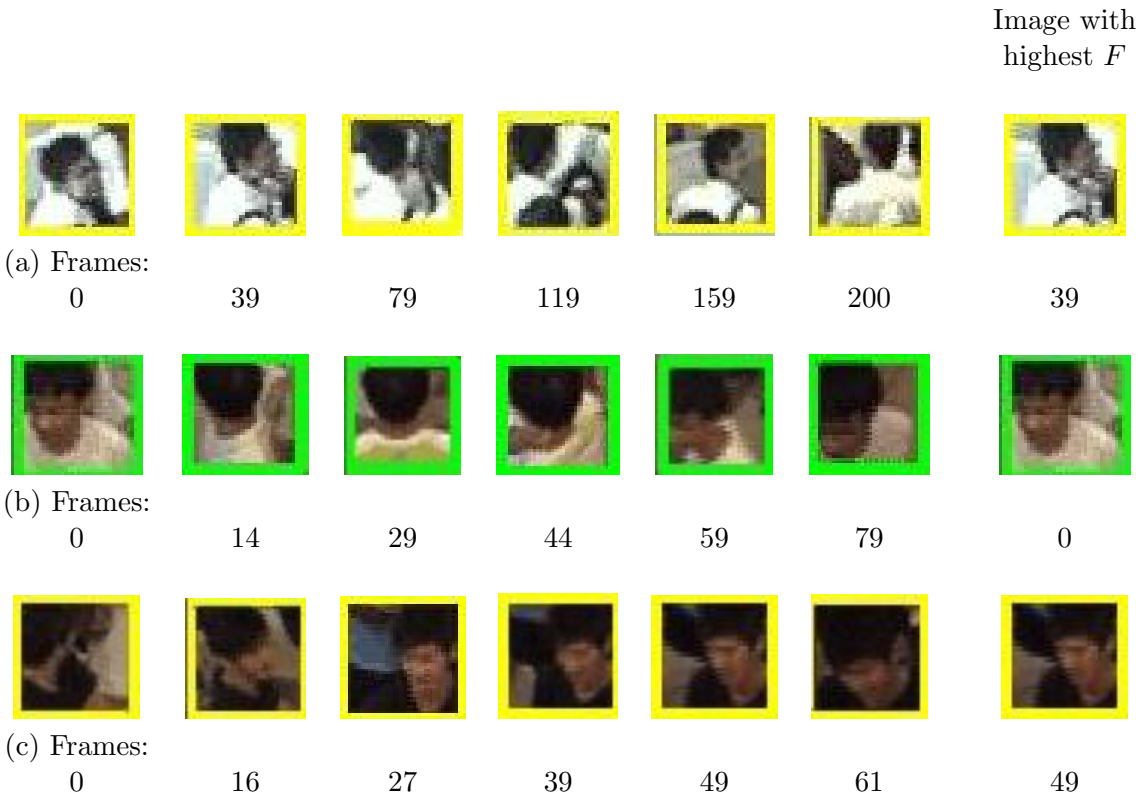


Figure 6.12: Finding the best frontal facial view of individuals in dense crowds: (a) yellow image window of Figure 5.10; (b) green image window of Figure 5.12; (c) yellow image window of Figure 5.13.

ure 6.12 are selected because they show significant changes in their head orientations throughout the video sequence. And from the results shown, our approach shows reasonably good performance even if applied to people in a dense crowd, given the slight displacements of the image windows from the tracked heads in some frames of the video sequence and the low resolution cropped images.

6.4 Concluding Remarks

We have presented an approach to finding the best frontal view of a person in a video sequence assuming that the head has been detected and tracked within an image window. To compute our proposed parameter, *Frontalness Value*, we used color information from the head region. This, unlike facial features, is not dependent on high image resolution. We have shown that our approach is promising with a high accuracy rate. Despite requiring no training, it is independent to person identity, robust with respect to occlusions and tolerant to facial image size as low as 20 x 24 pixels. Also, our approach works well under normal lighting variations in a room with complex backgrounds. Lastly, our experiments show the effectiveness of our approach when applied to tracked heads of individuals in dense crowd videos.

In comparison with [105], our approach defines two simple bounding boxes and is less complex computationally, although [105] also uses color information from facial images. Because of these bounding boxes, our approach has the potential to deal with faces under occlusion, with facial hair, or different hair styles, which will fail in [105]. On the other hand, our approach can be sensitive to the positioning of the image window which is occasionally displaced slightly during tracking, moving backgrounds as only simple frame differencing is applied here; and skin-like colors of non-skin regions from the background, clothing and headdress .

Chapter 7

Conclusions and Future Work

7.1 Conclusions

The main contribution of this thesis is in addressing an aspect of visual surveillance research, which is to detect and track individuals in dense crowds, that has largely been overlooked in the computer vision community despite increasing need for it. Novel techniques have been proposed in this thesis for detecting and tracking of individuals in dense crowds. Extensive experiments have been carried out to test algorithms in realistic scenarios. Lastly, we propose an application of detection and tracking: find the instance where the tracked face is most frontal. Like the proposed tracker, this method is also suitable for both single person and dense crowds scenarios.

Amongst feature-based detection approaches, temporal feature-based detection has the advantages of being non-object specific, being able to handle cluttered or complex backgrounds and moving camera, if the objects of interest are in rigid motion. However, we have found that a temporal feature-based approach is not suitable for detecting individuals in dense crowds. This is because of the combined head and shoulder articulations, where the assumption of rigid human motion fails. On the other

hand, the proposed spatial feature-based detection approach that learns a cascade of boosted classifiers using 2D Haar-like features, is not only able to provide detections of individuals in dense crowds, but is also able to segment their heads, making possible further processing like face recognition or behavior analysis. We have proposed novel techniques to increase the detection rate while maintaining the false alarm rate, or alternatively to reduce the false alarm rate while maintaining the detection rate of the detector. Our approach makes use of color information within the initially detected windows to build color bin images which are further classified as true detections or false alarms. In the other regression line approach, we use a weak perspective model of a single uncalibrated camera to further reduce the false alarm rate. Unlike other works, this approach only relies on the 2D image size and the 2D locations of the detections in images and does not require any 3D world information. Our only assumptions are that the people in the scene have the same 3D world size and the crowd distributes over a plane. An approximately linear relationship is found between image size and vertical location in the image. An algorithm based on least squares line fitting is then used to remove detections that are considered as false alarms (outliers).

Bayesian filtering, an optimal technique that uses all available information for object tracking, estimates the state (location, region, etc.) of the object from noisy measurements or observations. It provides a probabilistic framework for recursive state estimation, using the data-update and the prediction components. We propose a novel tracker in the Bayesian framework for tracking individuals in dense crowds (or also any initialized object in other scenarios). The algorithm uses several KLT feature points and computes their weights for tracking the object. In this framework, position vectors of the feature points are used to define the prior term and motion coherence of the feature points is used to define the likelihood term. In cases where there is significant object occlusion, the tracker switches to a linear approximation for object position. In

addition, the tracker incorporates methods to deal with scaling and rotation, and this improves the tracker's performance under such situations.

The final contribution of this thesis is to propose an approach to find the best frontal facial view of the detected and tracked head of an individual from the frames in a video sequence, so as to optimize the performance of any subsequent processing, such as face recognition. To compute the proposed measurement parameter, *Frontalness Value* in each image, color information from the head region is used instead of its facial features which is dependent on image resolution. The approach saves much computation, requires no training and is independent of person identity, robust with respect to occlusions and tolerant to different facial image sizes.

7.2 Future Work

The current proposed head detector in dense crowds is trained and tested with Asian crowd scenes. Like any other machine learning algorithm, the Viola-Jones and the color bin image approaches of the detector are biased towards their training samples. Hence, the performance of the current proposed head detector may be easily challenged by Caucasian heads with color intensities that differ significantly from the positive training samples. Future work should include data from non-Asian crowds, so that both the Viola-Jones and the color bin image approaches can be improved to handle more heterogeneous dense crowds. This implies using more 2D Haar-like features for the training of both the initial Viola-Jones head detector and the color bin image classifier.

The proposed tracker's strength is its non-object specificness and it performs well in cluttered or complex backgrounds. During occlusions, the proposed linear approximator may fail if the targeted object moves at a different velocity during occlusion. Also, in other challenging scenarios such as when another object merges with the

tracked object and then splits, the tracker will have difficulty maintaining the correct track. Therefore, a hybrid approach that combines both the head detector and the tracker, can be investigated to build a more robust tracker which specifically tracks heads within dense crowds.

Bibliography

[1] M. Weiser. Some computer science issues in ubiquitous computing. *Communications of the ACM*, 36(7):75–84, July 1993.

[2] R. Cipolla and A. Pentland. *Computer-Vision for Human-Machine Interaction.* Cambridge University Press, Cambridge, 1998.

[3] M. Turk and G. Robertson. Perceptual user interfaces. *Communications of the ACM (special issue)*, 43(3):32–70, March 2000.

[4] M. Valera and S. A. Velastin. Intelligent distributed surveillance systems: a review. *IEE Proceedings on Vision, Image, and Signal Processing*, 152(2):192–204, April 2005.

[5] W. Hu, T. Tan, L. Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *IEEE Transactions on Systems, Man and Cybernatics*, 34(3):334–352, August 2004.

[6] The Defense Advanced Research Projects Agency (DARPA). http://www.darpa.mil/.

[7] BBC News, United Kingdom: CCTV boom 'failing to cut crime'. http://news.bbc.co.uk/1/hi/uk/7384843.stm.

[8] L.M. Fuentes and S.A. Velastin. People tracking in surveillance applications. *Image and Vision Computing*, 24(11):1165–1171, November 2006.

[9] J. Heikkil and O. Silvn. A real-time system for monitoring of cyclists and pedestrians. *Image and Vision Computing*, 22(7):563–570, July 2004.

[10] C. Kim and J-N. Hwang. Object-based video abstraction for video surveillance systems. *IEEE Transactions on Circuits and Systems for Video technology*, 12(12):1128–1138, December 2002.

[11] I. Haritaoglu, D. Harwood, and L. S. Davis. W4: Real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):809–830, August 2000.

[12] Y. Huang and I.A. Essa. Tracking multiple objects through occlusions. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1051–1058, 2005.

[13] P. Kumar, S. Ranganath, K. Sengupta, and W.M. Huang. Cooperative multitarget tracking with efficient split and merge handling. *IEEE Transactions on Circuits and Systems for Video technology*, 16(12):1477–1490, December 2006.

[14] T. Darrell and A. Pentland. Space-time gestures. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 335–340, 1993.

[15] N. T. Nguyen, H. H. Bui, S. Venkatesh, and G. A. W. West. Recognising and monitoring high-level behaviours in complex spatial environments. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 620–625, 2003.

[16] Y. A. Ivanov and A. F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):852–872, August 2000.

[17] W. Hu, D. Xie, and T. Tan. A hierarchical self-organizing approach for learning the patterns of motion trajectories. *IEEE Transactions on Neural Networks*, 15(1):135–144, January 2004.

[18] J.R. Casas, A.P. Sitjes, and P.P. Folch. Mutual feedback scheme for face detection and tracking aimed at density estimation in demonstrations. *Vision, Image and Signal Processing*, 152(3):334–346, June 2005.

[19] V. Rabaud and S. Belongie. Counting crowded moving objects. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 705–711, 2006.

[20] S. J. Mckenna, S. Jabri, Z. Duric, H. Wechsler, and A. Rosenfeld. Tracking groups of people. *Computer Vision and Image Understanding*, 80(1):42–56, October 2000.

[21] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–577, May 2003.

[22] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: active contour models. *International Journal of Computer Vision*, 1(4):321–331, January 1988.

[23] N. Paragios and R. Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(3):266–280, March 2000.

[24] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(4):349–361, April 2001.

[25] N. Peterfreund. Robust tracking of position and velocity with Kalman snakes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(6):564–569, June 1999.

[26] I.A. Karaulova, Hall P.M., and Marshall A.D. A hierarchical model of dynamics for tracking people with a single video camera. In *British Machine Vision Conference*, pages 352–361, 2000.

[27] Q. Delamarre and O. Faugeras. 3d articulated models and multiview tracking with physical forces. *Computer Vision and Image Understanding*, 81(3):328–357, March 2001.

[28] R. Plankers and P. Fua. Articulated soft objects for multiview shape and motion capture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1182–1187, September 2003.

[29] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 878–885, 2005.

[30] E. Seemann, M. Fritz, and B. Schiele. Towards robust pedestrian detection in crowded image sequences. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, 2007.

[31] T. Zhao and R. Nevatia. Tracking multiple humans in crowded environment. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 406–413, 2004.

[32] T. Zhao and R. Nevatia. Tracking multiple humans in complex situations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1208–1221, September 2004.

[33] B. Wu and R Nevatia. Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors. *International Journal of Computer Vision*, 75(2):247–266, November 2007.

[34] R. Polana and R. Nelson. Low level recognition of human motion (or how to get your man without finding his body parts). In *IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pages 77–82, 1994.

[35] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik. A real-time computer vision system for vehicle tracking and traffic surveillance. *Transportation Research Part C: Emerging Technologies*, 6(4):271–288, August 1998.

[36] T.-J. Fan, G. Medioni, and R. Nevatia. Recognizing 3-d objects using surface descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1140–1157, November 1989.

[37] D. S. Jang and H. I. Choi. Active models for tracking moving objects. *Pattern Recognition*, 33(7):1135–1146, July 2000.

[38] H. Liu, N. Dong, and H. Zha. Omni-directional vision based human motion detection for autonomous mobile robots. In *IEEE International Conference on Systems, Man and Cybernatics*, volume 3, pages 2236–2241, 2005.

[39] A. J. Lipton, H. Fujiyoshi, and R. S. Patil. Moving target classification and tracking from real-time video. In *IEEE Workshop Applications of Computer Vision*, pages 8–14, 1998.

[40] Fleet D.J. Barron, J.L. and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[41] H. Tsutsui, J. Miura, and Y. Shirai. Optical flow-based person tracking by multiple cameras. In *International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 91–96, 2001.

[42] D. Meyer, J. Denzler, and H. Niemann. Model based extraction of articulated objects in image sequences for gait analysis. In *IEEE International Conference on Image Processing*, volume 3, pages 78–81, 1997.

[43] D. Meyer, J. Posl, and H. Niemann. Gait classification with hmms for trajectories of body parts extracted by mixture densities. In *British Machine Vision Conference*, pages 459–468, 1998.

[44] H. Sidenbladh, M.J. Black, and D.J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *European Conference on Computer Vision*, pages 702–718, 2000.

[45] H. Ning, L. Wang, W. Hu, and T. Tan. Articulated model based people tracking using motion models. In *IEEE International Conference on Multimodal Interfaces*, pages 383–388, 2002.

[46] Q. Zhu, M.C. Yeh, K.T. Cheng, and S. Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1491–1498, 2006.

[47] J. Rittscher, P.H. Tu, and N. Krahnstoever. Simultaneous estimation of segmentation and shape. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 486–493, 2005.

[48] O. Tuzel, F.M. Porikli, and P. Meer. Human detection via classification on riemannian manifolds. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, 2007.

[49] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust part detectors. In *European Conference on Computer Vision*, volume 1, pages 69–82, 2004.

[50] Ec funded caviar project / ist 2001 37540. http://homepages.inf.ed.ac.uk/rbf/caviar/.

[51] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, October 2005.

[52] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[53] C. Tomasi and T. Kanade. *Detection and Tracking of Point Features*. PhD thesis, Carnegie Mellon University Technical Report CMU-CS-91-132, April 1991.

[54] J. Shi and C. Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[55] G.J. Brostow and R Cipolla. Unsupervised bayesian detection of independent motion in crowds. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 594–601, 2006.

[56] KLT: An Implementation of the Kanade-Lucas-Tomasi Feature Tracker. http://www.ces.clemson.edu/ stb/klt.

[57] K. Kanatani and Y. Sugaya. Multi-stage optimization for multi-body motion segmentation. In *Australia-Japan Advanced Workshop on Computer Vision*, pages 25–31, 2003.

[58] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (GPCA). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1945–1959, December 2005.

[59] D. Hoiem, A.A. Efros, and M. Hebert. Putting objects in perspective. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2137–2144, 2006.

[60] C-H. Sim and S. Ranganath. Reducing false alarms for detections in crowd. In *Asian Conference on Computer Vision Workshop on Multi-dimensional and Multi-view Image Processing*, pages 164–169, 2007.

[61] P.A. Viola and M.J. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 511–518, 2001.

[62] P. Sabzmeydani and G. Mori. Detecting pedestrians by learning shapelet features. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[63] P. Dollar, Z.W. Tu, H. Tao, and S. Belongie. Feature mining for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[64] S. Munder and D. Gavrila. An experimental study on pedestrian classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1863–1868, November 2006.

[65] H.J. Joo, B.W. Jang, S. Suman, and P.K. Rhee. Use of nested k-means for robust head location in visual surveillance system. In *Pacific Rim International Conference on Artificial Intelligence*, pages 583–592, 2006.

[66] Y-G. Kim, J-E. Lee, S-J. Kim, S-M Choi, and G-T. Park. Head detection of the car occupant based on contour models and support vector machines. In *Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, pages 59–61, 2005.

[67] M. Clabian, H. Rtzer, H. Bischof, and W. Kropatsch. Head detection and localization from sparse 3d data. In *Proceedings of the 24th DAGM Symposium on Pattern Recognition*, pages 395–402, 2002.

[68] Y. Li, H. Ai, C. Huang, and S. Lao. Robust head tracking based on a multi-state particle filter. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 335–340, 2006.

[69] Y. Jin and F. Mokhtarian. Towards robust head tracking by particles. In *IEEE International Conference on Image Processing*, volume 3, pages 864–867, 2005.

[70] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 886–893, 2005.

[71] Picasa web albums: Fast and easy photo sharing from google. http://picasaweb.google.com/merv.mel/indoorgamescompetition/.

[72] R. Lienhart and J. Maydt. An extended set of Haar-like features for rapid object detection. In *IEEE International Conference on Image Processing*, volume 1, pages 900–903, 2002.

[73] J.P. Renno, J. Orwell, and G.A. Jones. Learning surveillance tracking models for the self-calibrated ground plane. In *British Machine Vision Conference*, 2002.

[74] J.W. Wisnowski, D.C. Montgomery, and J.R. Simpson. A comparative analysis of multiple outlier detection procedures in the linear regression model. *Computational Statistics and Data Analysis*, 36(3):351–382, May 2001.

[75] P. Meer, D. Mintz, A. Rosenfeld, and D.Y. Kim. Robust regression methods for computer vision: A review. *International Journal of Computer Vision*, 6(1):59–70, April 1991.

[76] G. Bradski, A. Kaehler, and V. Pisarevsky. Learning-based computer vision with Intel's open source computer vision library. *Intel Technology Journal*, 9(2):118–131, May 2005.

[77] INRIA Object Detection and Localization Toolkit. http://pascal.inrialpes.fr/soft/olt/.

[78] P.A. Viola and M.J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, May 2004.

[79] Flickr: Share your photos. watch the world. http://www.flickr.com/.

[80] D. Fox, J. Hightower, Liao. L., D. Schulz, and G. Borriello. Bayesian filtering for location estimation. *IEEE Pervasive Computing*, 02(3):24–33, July-September 2003.

[81] The Kalman Filter homepage. http://www.cs.unc.edu/∼welch/kalman/.

[82] E. Brookner. *Tracking and Kalman Filtering Made Easy.* John Wiley & Sons, 1998.

[83] The Condensation Algorithm homepage. http://www.robots.ox.ac.uk/∼misard/condensation.html.

[84] M. Isard and A. Blake. CONDENSATION-conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, August 1998.

[85] D.S. Tweed and A.D. Calway. Tracking multiple animals in wildlife footage. In *International Conference on Pattern Recognition*, volume 2, pages 24–27, 2002.

[86] Y. Huang, T.S. Huang, and H. Niemann. Segmentation-based object tracking using image warping and Kalman filtering. In *IEEE International Conference on Image Processing*, volume 3, pages 601–604, 2002.

[87] R. Kumar, H.S. Sawhney, S. Samarasekera, S. Hsu, H. Tao, Y.L. Guo, K. Hanna, A. Pope, R. Wildes, D. Hirvonen, M. Hansen, and P. Burt. Aerial video surveillance and exploitation. *Proceedings of the IEEE*, 89(10):1518–1539, October 2001.

[88] G. Peters, C. Eckes, and C. von der Malsburg. Tracking of rotating objects. In *Advances in Pattern Recognition*, pages 390–396, 1998.

[89] S.D. Tran and L.S. Davis. Robust object tracking with regional affine invariant features. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007.

[90] S. Ali and M. Shah. A Lagrangian particle dynamics approach for crowd flow segmentation and stability analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6, 2007.

[91] University of Central Florida Sports Action Dataset. http://server.cs.ucf.edu/ vision/data.html.

[92] K. Okada and C. von der Malsburg. Analysis and synthesis of human faces with pose variations by a parametric piecewise linear subspace method. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 761–768, 2001.

[93] J. Sherrah and S. Gong. Fusion of perceptual cues for robust tracking of head pose and position. *Pattern Recognition*, 34(8):1565–1572, August 2001.

[94] J.G. Wang and E. Sung. Pose determination of human faces by using vanishing points. *Pattern Recognition*, 34(12):2427–2445, December 2001.

[95] G. Liang, H. Zha, and H. Liu. Affine correspondence based head pose estimation for a sequence of images by using a 3d model. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 632–637, 2004.

[96] S.Z. Li and Z.Q. Zhang. Floatboost learning and statistical face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1112–1123, September 2004.

[97] S. Srinivasan and K.L. Boyer. Head pose estimation using view based eigenspaces. In *International Conference on Pattern Recognition*, volume 4, pages 302–305, 2002.

[98] N. Kruger, M. Potzsch, and C. von der Malsburg. Determination of face position and pose with a learned representation based on labeled graphs. *Image and Vision Computing*, 15(8):665–673, August 1997.

[99] R. Ishiyama and S. Sakamoto. Fast and accurate facial pose estimation by aligning a 3d appearance model. In *International Conference on Pattern Recognition*, volume 4, pages 388–391, 2004.

[100] Q. Ji and R. Hu. 3d face pose estimation and tracking from a monocular camera. *Image and Vision Computing*, 20(7):499–511, May 2002.

[101] J. Xiao, T. Moriyama, T. Kanade, and J.F. Cohn. Robust full-motion recovery of head by dynamic templates and re-registration techniques. *International Journal of Imaging Systems and Technology*, 13(1):85–94, 2003.

[102] F. Fleuret and D. Geman. Fast face detection with precise pose estimation. In *International Conference on Pattern Recognition*, volume 1, pages 235–238, 2002.

[103] S. Malassiotis and M.G. Strintzis. Robust real-time 3d head pose estimation from range data. *Pattern Recognition*, 38(8):1153–1165, August 2005.

[104] D. Chai and K.N. Ngan. Face segmentation using skin-color map in videophone applications. *IEEE Transactions on Circuits and Systems for Video technology*, 9(4):551–564, June 1999.

[105] Q. Chen, H. Wu, T. Fukumoto, and M. Yachida. 3d head pose estimation without feature tracking. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 88–93, 1998.

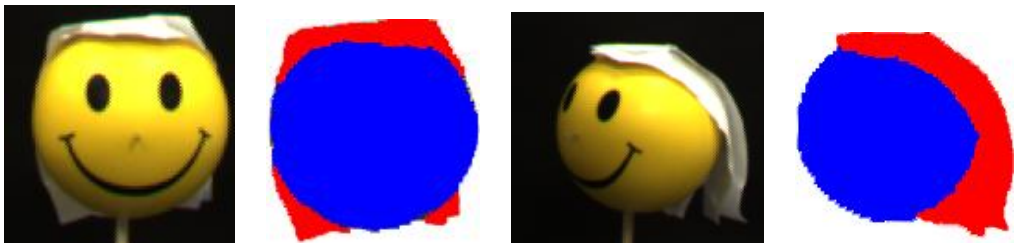
Appendix A

Validation of *Frontalness Value*

Here, we conduct experiments to validate the definition of the *Frontalness Value* in Step 4 of Section 6.2.1, using only one camera.

Experiments

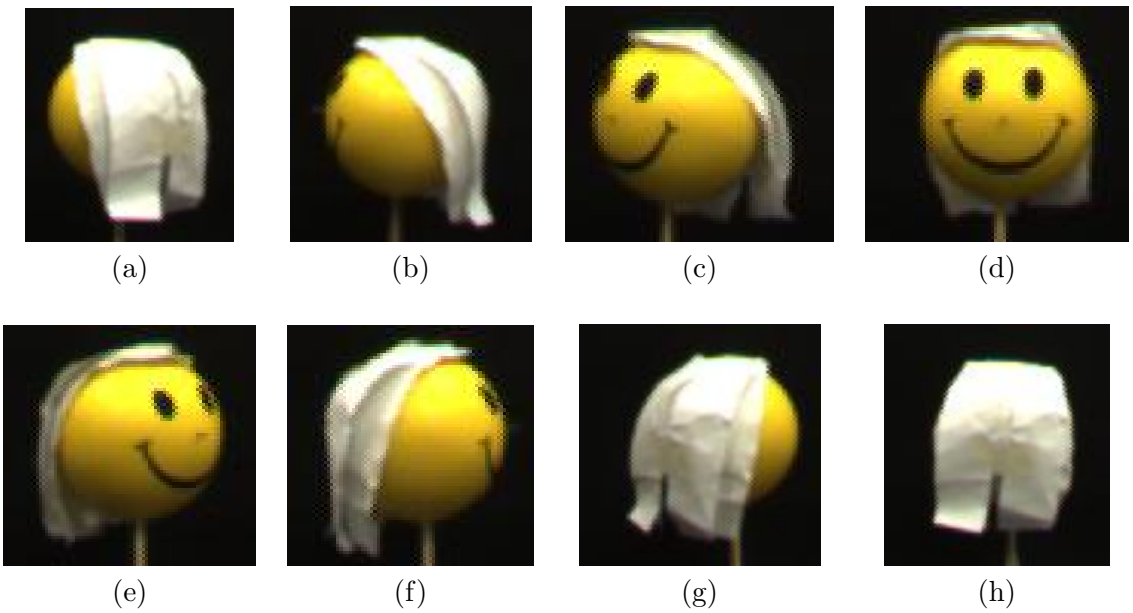
For the experiments, we manually extract skin color and hair color from a model, as in Figure A.1, since our approach is sensitive to the performance of combining frame differencing with skin color detection.



(a) (b) (c) (d)

Figure A.1: Image examples of the model faces and their manually processed images: (a) Frontal face view of the model; (b) Processed image of (a) after manually detecting its skin and hair color; (c) Another face view of the model; (d) Processed image of (c) after manually detecting its skin and hair color.

Then, we perform steps 2 to 4 of Section 6.2.1 to the manually processed images, as in Figure A.1(b) and A.1(d), such that the model is captured at eight different orientations (i.e. 0°, ±45°, ±90°, ±135°, +180°) for all of the eight different image resolutions (i.e. 106 x 106, 77 x 77, 53 x 53, 44 x 44, 32 x 32, 26 x 26, 18 x 18, 14 x 14), giving a total of 64 sets of *Skin Area Ratio*, *Horizontal Deviation from Center* and *Frontalness Values*. See Figure A.2 for the raw data input images for 53 x 53 pixels resolution.



(a) (b) (c) (d)

(e) (f) (g) (h)

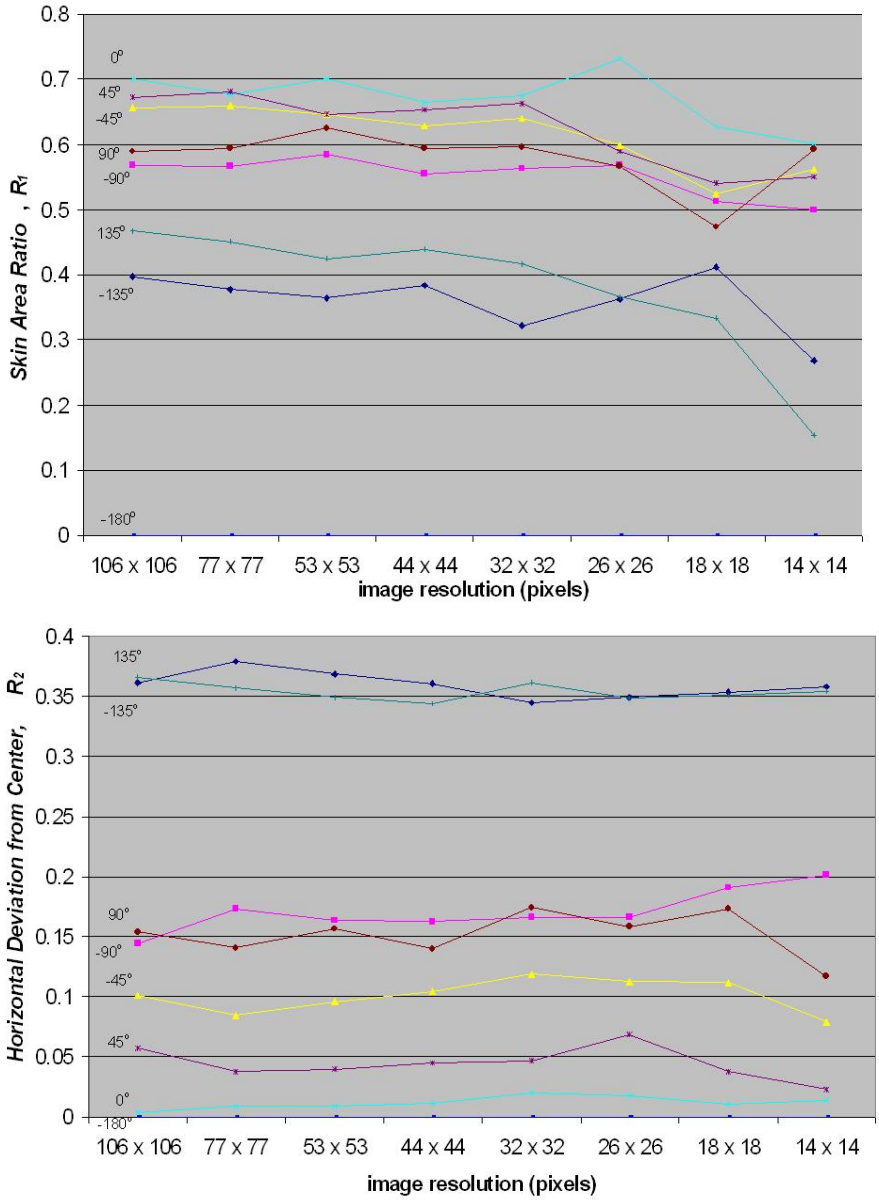
Figure A.2: Face views of model for 53 x 53 pixels resolution: (a) −135°; (b) −90°; (c) −45°; (d) 0°; (e) +45°; (f) +90°; (g) +135°; (h) +180°.

Results and Discussions

The computed *Skin Area Ratio*, R_1 and *Horizontal Deviation from Center*, R_2 from the 64 different input data images are plotted on the graphs shown in Figure A.3. Notice that both R_1 and R_2 values for each orientation do not vary much at high image resolutions, but they become more and more unstable at low image resolutions.

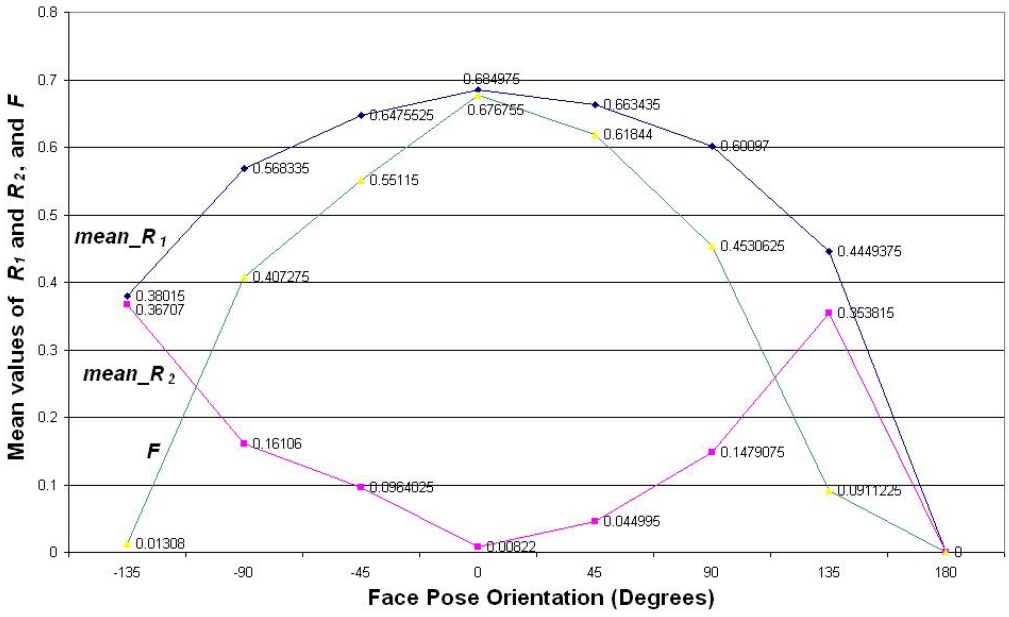
To validate the definition of the *Frontalness Value*, F, we use only the stable portion of the graphs. Hence, only the first four sets of R_1 and R_2 from high image resolutions (i.e. 106 x 106, 77 x 77, 53 x 53, 44 x 44) are used to calculate the mean values for R_1 and R_2. See Figure A.4.

From the graph in Figure A.4 where $F = mean\_R_1 - mean\_R_2$, we see that F peaks at 0° which is the frontal pose, and decreases as the face pose deviates from 0°, almost at a constant rate. This validates that choosing the maximum F from among the multiple camera views, for which the definition of $F = R_1 - R_2$, corresponds to the camera view with face pose closest to the frontal pose. Moreover, F gives a conical shape graph that has a steeper gradient than R_1 and R_2 in terms of magnitude, therefore F is a better decision maker than R_1 or R_2 for finding the best frontal face pose.



(a) (b)

Figure A.3: Plotted R_1 and R_2 values of the model at different orientations in different image resolutions.



(a)

Figure A.4: Plotted mean values of R_1 and R_2, and their corresponding F for each of the eight orientations.